Proceedings of the 5th WSEAS International Conference on Telecommunications and Informatics, Istanbul, Turkey, May 27-29, 2006 (pp41-45)

Study on Role-Based Access Control Model for Web Services and its Application

MIN WU, JIAXUN CHEN, YONGSHENG DING
College of Information Sciences and Technology
Donghua University
1882 Yan'an Road (W), Shanghai 200051
CHINA

*Abstract***.** According to the advantage of platform independent, loose coupling and opening, Web Services application infrastructure becomes preferred solution to enterprise information sharing and enterprise application integration. New business applications can be dynamically assembled from a variety of Web Services. However, it also presents challenges in terms of security and management. Large-scale interconnection of systems and services, decentralized administration, and rapidly changing service compositions require a flexible access control model that is adapted to these challenges. In this paper, we focus on access control of Web Services based on its security requirements. We propose model and mechanism for specifying and enforcing role-based authorization models for Web Services. We also develop a prototype application to demonstrate the practical feasibility of this technology.

Key-Words: Web Services, Security, Access Control, Role-Based Access Control

1 Introduction

The evolution of business technology leads to collaborative systems that can be dynamically integrated to instantly deliver business value. Web Services technology allows loosely coupled applications to be assembled as services and distributed over a connected information technology infrastructure. Through the use of protocols such as XML, SOAP, WSDL, and UDDI, applications can more easily be communicated with each other, enabling faster and cheaper enterprise application integration. Information systems serving organizations such as universities, hospitals, and government agencies must handle large numbers of users and must be easy to use, internally efficient, secure, reliable, and robust. Moreover, the secure interoperability of independent services is also important. The emerging proliferation of Web Services also presents challenges in terms of security and management.

According to its protocol stacks, security and management of Web Services consists of several layers [1]. Lower level security aims to assure the secure data transmission, using cryptography and communication security mechanisms. Higher level security addresses to provide a flexible access control management□which comprises identity management, authentication, authorization, audit, trust etc. Highest level security is Web Services composition, which aims to management of Web Services, such as routing, provisioning, etc.

The dynamic, scale, flexibility, and interoperability of the underlying services oriented architecture require an access control model that is adapt to those characteristic. A number of standards, such as XML Encryption [2] and WS-Security [3] have already emerged to address the problem of secure transmission concern data integrity, non-repudiation, and encryption. Efforts that solve problem regarding

the access control for Web Services are just at the beginning, such as, the eXtensible Access Control Markup Language (XACML) [4] and Security Assertions Markup Language (SAML) [5] are two visible efforts. Mature access control mechanism and the reusable components are lack and the enforcement of access control is always re-inventing the wheel [6].

View from the security mechanism of the role-based access control (RBAC) model [7], it should meet the requirements of Web Services access control. The RBAC is characterized by the notion that permissions are assigned to roles, and not directly to users. Users are assigned to appropriate roles according to their job functions, and hence indirectly acquire the permissions associated with those roles. In this paper, we will present a RBAC-based access control model for Web Services (RBAC_WS) and demonstrate its flexibility in a student grade management application.

The remainder of this paper is organized as follows: Section 2 briefly compares the related work. Section 3 presents the RBAC_WS. Section 4 provides the implementation of prototype system. Finally, we give our conclusions in Section 5.

2 Related work

Because of the characters of Web Services and the complexity of the distributed environment, its security is a large challenge problem. Damiani [8] proposes the notion of a fine-grained access control model on XML document, by exploiting XML own capability. Bhatti [9] proposes X-RBAC, an XML-based RBAC policy specification framework for enforcing access control in dynamic XML based Web Services. But they focus on controlling access to XML document, not exactly to Web service. Nakamur [10] proposes an access control system for Web Services and discusses how security information sent with SOAP message. But it has platform limitation of J2EE. Although some researchers [11, 12] discuss the security of SOAP messages, they do not provide access control model for Web Services

To the best of our knowledge, no complete RBAC access control model framework for Web Services has been reported. Web Services can employ RBAC to manage increasingly large number of users securely. As a result, users gain access to only that information they need to complete their jobs, and privacy of data is protected.

3 Role-Based Access Control Model for Web Services

3.1 Requirements of Web Services Access Control

The access control of Web Services includes two main processes referred to as authentication and authorization. Authentication defines how to establish identity. Authorization permits or denies that identity to access resources.

Web Services are programmatic interfaces, as thus it is hard to monitor its suspicious activity. For instance, a user in Web Services application has his interface. Whereas a request for other information would raise immediate suspicion from an administrator, but access to an improperly protected SOAP interface can easily go undetected. Because the user will have access to more and more services and without human checkpoints, access rights should be actively managed using “least privilege” principles in order to improve control. In addition, multiple administrators with different access rights should be considered. Having a single administrator with all access rights is a single point of failure, and will be very hard to detect if the administrator is the compromise. Having multiple administrators can help spread risk and provide further checks and balances. Moreover, it should provide accountability that administrators have view only access to logging and audit data on their activities.

An access control scheme based on an access matrix, in which each entry specifies a given user’s rights to invoke a particular Web Services, is inadequate. It neither scales for large numbers of Web Services and users nor captures the relationship of services. An

access control architecture should impose as little as possible on each service while allowing secure interaction between services. Services running in different management domains need mechanisms that allow them to negotiate and interoperate.

Through the above analyses, we identified the following additional requirements to access control of Web Services: 1) Least privilege; 2) Easy management; 3) Delegation of duties; 4) Separation of duties; 5) Fine grained authorization; and 6) Scalability. Hence, the RBAC_WS should consider above requirements.

3.2 RBAC_WS Model

From the RBAC96 model [13], we extend it to a RBAC model for Web Services (RBAC_WS) as shown in Fig. 1.

The definitions of the elements of the RBAC_WS in Fig. 1 are as follows:

AT: Access Types, including execute, modify and query□

SO: Web Services Objects;

SC: Web Services Collection□

SM: Web Services Method;

IM (Mapping): Mapping between SC and SO□

MM (Mapping): Mapping between SO and SM;

WSCH: Web Services Collection Hierarchy;

$CP \subseteq AT \times SC$: Web Services Collection based Permission;

$OP \subseteq AT \times SO$: Web Services Object based Permission;

$MP \subseteq AT \times SM$: Web Services Method based Permissions;

$P = CP \cup OP \cup MP$: Permissions;

$CPA \subseteq R \times CP$: Role-Collection Permission Assignment;

$OPA \subseteq R \times OP$: Role-Object Permission Assignment□

$MPA \subseteq R \times MP$: Role-method Permission Assignment□

$PA = CPA \cup OPA \cup MPA$: Permission Assignment between Role and Web Services.

In the RBAC_WS, the Users, Roles, Role Hierarchy, User-Role Assignment and Session are the same as those of the RBAC96 model. However, different from direct assignment between roles and permission, there are *P* and *PA* in the RBAC_WS. *P* is the permission aggregation within Web Services Collection, Web Services Object and Web Services Method. *PA* is the permission assignment between roles and Web Services including Web Services Collection, Web Services Object and Web Services Method. By the *IM* mapping function from *SC* to *SO*, *CP* and *CPA* imply the *OP* and *OPA*. Also by the *MM* mapping function from *SO* to *SM*, *OP* and *OPA* imply the *MP* and *MPA*. The permissions will be transferred to high level collections from low level according to Web Services Collection Hierarchy.

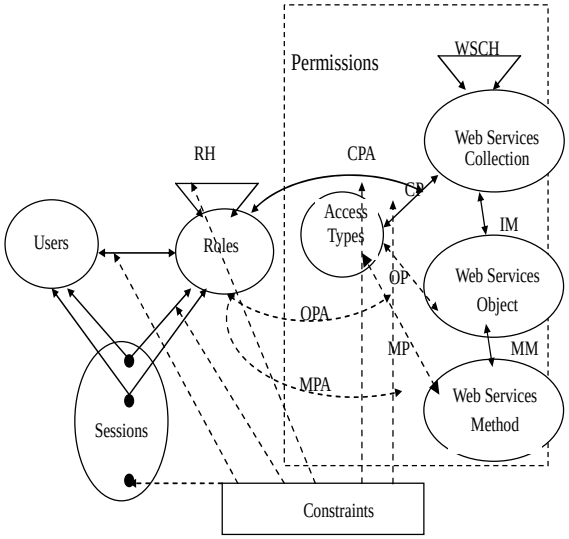

Fig. 1. The RBAC_WS model

3.3 Permission Assignment Mechanism

With the fast transferring in business processes of enterprise, alteration to the operation of Web Services, such as adding, deleting and modifying is frequent. Since alteration of the user-role assignment (URA) is less than the alteration of the role-permission assignment (PRA), in our RBAC_WS model, the permission assignment is subject to Web Services. We give access control permission definition in a 5-*tuple* (*object operator action type Cons*) as follows:

1) *Object* refers to the object which can be accessed, including Web Services collection, Web Services

object and Web Services method.

2) *Operator (userid, role_domain, roleid)* refers to requester set, where role_domain is the domain that role is belong to.

3) *Action (execute modify query)* defines the operation set to Web Services, including execute, modify and query.

4) *Type* defines whether authorizations propagation is permitted. If the value of type is "deny", the propagation is denied. Otherwise, the propagation is permitted.

5) *Cons* defines constraint set.

According to the above definitions, authorizations may be applied to Collections, and could be propagated to Web Services Collections and Web Services Objects of its children. However, sometimes propagation is not allowed to Web Services Objects. In such case, the parameter *type* could be used to treat with: When the value of *type* is "deny", propagation is not allowed to Web Services Objects, but permitted to children Web Services Collection down the hierarchical trees structure. Otherwise, propagation is permitted to Web Services Collections and Web Services Objects of its children.

4 An Application Example

In order to examine the practical feasibility of the RBAC_WS, we establish a prototype application to student grade management system. Assuming there are two kinds of services, i.e., admin management service and grade management service. Admin management service provides a function of maintaining user table and grade table, while Grade management service includes services of View_Grade, Edit_Grade, and Delete_Grade. Three roles including student, teacher, and admin have different operation permissions to access user table and grade table, as shown in Table 1. The logical implementation architecture of access control is shown in Fig. 2.

Table 1. Role and permission of student grade management system

Function / Role	View_ Grade	Edit_ Grade	Delete_ Grade	Maintain User and Grade
Student	√	x	x	x
Teacher	√	√	√	x
Admin	√	√	√	√

We use Tomcat 5.0 and JSP pages for test portal. The Web Services is published using Apache Axis framework [14]. This framework is based on Java Web Technology (Servlet). We use an Axis Handler to perform authentication and authorization control. The data (service definition, user definition, permission definition) is stored in a Mysql database. Because of our motivation for integration with open standards, we use SAML encoding for representing user authentication, user-role assignment, and permission-role assignment.

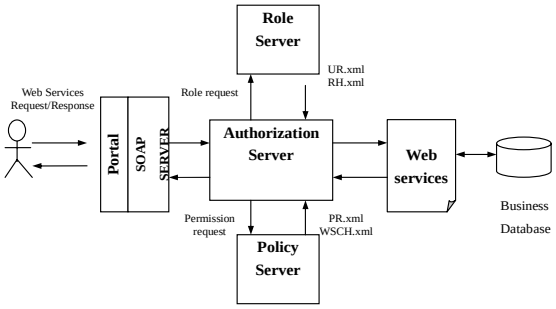

Fig. 2. Logical implementation architecture of access control

Take a teacher user as an example, we describe the working process as follows.

1) User (a teacher) logging in portal;
2) Portal capturing user's authentication and authorization credentials;
3) Portal creating and signing SAML assertion, and placing SAML in a SOAP message;
4) Portal sending SOAP message to Web Services;
5) Web Services Handler accepting or denying request to Web Services based on original user's role (RBAC);
6) Web Services Handler finally sending message to Web Services;
7) Web Services processing;

8) Web Services sending response back to portal.

The implementation of prototype application proves that our RBAC_WS meets with the main access control requirements mentioned in Section 3.1. RBAC_WS model allocates privileges to roles, and it is much easier to administrate the privileges to a limited number of roles, such as teacher, student, and admin. According to Web Services collection hierarchies, we can define least privileges to the junior roles (student) and the superior roles (teacher) inherit the privileges of the subordinate roles, as well as having their own additional privileges. It supports the fine grained authorization and easy management.

5 Conclusions

In this paper, we developed model, architecture, and mechanism for specifying and enforcing role-based authorization models for Web Services. We also developed a prototype to demonstrate the practical feasibility of this technology. Next, based on RBAC_WS, we will apply XML Encryption and XML Digital Signature standards to provide message confidentiality and authenticity, respectively. And, the federated identity management of Web Services should be considered.

Acknowledgements

This work was supported in part by the National Nature Science Foundation of China (No. 60474037), and Program for New Century Excellent Talents in University (No. NCET-04-415).

References:

[1] The Web Services Protocol Stack. *Http://roadmap.cbdiforum.com/reports/protocols/*

[2] W3C XML Encryption Working Group. *Http://www.w3.org/Encryption/2001/*

[3] Web Services Security (WS-Security) Specification.*Http://www-106.ibm.com/developer-works/webservices/library/ws-secure/*

[4] XACML 1.0 Specification. *Http://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf*

[5] Security Assertions Mark-up Language (SAML) OASIS. XML-Based Security Services Technical Committee.*http://www.oasis-open.org/committees/security/*

[6] T. Ziebermayr, S. Probst: Web Service Authorization Framework, *Proceedings of the IEEE International Conference on Web Services*. San Diego, California, United States, July 6-9, 2004

[7] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, R. Chandramouli: Proposed NIST Standard for Role-based Access Control. ACM *Transactions on Information and System Security*, 4(224-274), 2001

[8] E. Damiani, S.Vimercati, S. Paraboschi, P. Samarati, Design and implementation of an access control processor for XML documents. *Computer Networks*, v.33 n.1-6, June 2000,pages 59-75.

[9] R. Bhatti, J. B. D. Joshi, E. Bertino, A. Ghafoor, Access Control in Dynamic XML-based Web-Services with XRBAC, In proceedings of *The First International Conference on Web Services*, Las Vegas, June 23-26, 2003

[10] Y. Nakamur, S. Hada, and R. Neyama, Towards the Integration of Web Services Security on Enterprise Environments, *Symposium on Applications and the Internet Workshops*, Narar City, Nara, Japan, 2002

[11] Web Services Security version1.005, April 2002. *http://www-06.ibm.com/developerworks/webservices/library/ws-secure/*

[12] Web Services Security Core Specification Working Draft 01, 20 September 2002. *http://lists.oasis-open.org/archives/wss/200209/pdf00000.pdf*

[13] R. Sandhu, E. Coyne, H. Feinstein, C. Youman: Role-Based Access Control Models. *IEEE Computer*, 29(38-47), 1996

[14] Web Services- Axis. *Http://ws.apache.org/axis/*