한국산학기술학회논문지
Vol. 8, No. 6, pp. 1394-1401, 2007

A Design and Implementation of Java Library for Multiple Access Control Models

Se-Jong Oh[1*]

다중 접근제어 모델을 위한 Java 라이브러리의 설계 및 구현

오세종[1*]

Abstract Secure access control is a hot issue of large-scale organizations or information systems, because they have numerous users and information objects. In many cases, system developers should implement an access control module as a part of application. This way induces difficult modification of the module and repeated implementation for new applications. In this paper we implement a Java API library for access control to support system developers who use Java. They can easily build up access control module using our library. Our library supports typical access control models, and it can offer new types of access control. Furthermore, it is able to run multiple access control models at the same time.

Key Words : Security, Access control, Java, Java package

요 약 안전한 접근제어는 많은 수의 사용자와 정보자원을 가지고 있는 대규모의 조직, 정보시스템에서 매우 중요한 관심사이다. 많은 경우에 시스템 개발자들은 응용 시스템의 일부로서 접근제어 모듈을 구현해야 한다. 이러한 방식은 접근제어 모듈의 수정을 어렵게 하고, 새로운 응용 시스템을 개발할 때 마다 반복적으로 접근제어 모듈을 개발해야 하는 문제가 있다. 본 논문에서는 Java를 사용하는 개발 환경을 위한 접근제어 Java API 라이브러리의 구현에 대해 제시한다. 개발자들은 제안된 라이브러리를 활용하여 쉽게 접근제어 모듈을 구현할 수 있다. 제안된 라이브러리는 주요 접근제어 모델뿐만 아니라 도메인 기반, 객체 중심의 접근제어와 같은 새로운 모델도 지원한다. 또한 여러 접근제어 모델을 동시에 적용할 수 있는 기능을 포함하고 있다.

1. Introduction

Many people understand that security in information system means to protect the system from outside malicious attackers. However contemporary information systems need to control inner authorized users' access according to their authority range, because they have numerous users and system resources, and there may exist information outflow by inner users. Furthermore, developing web-based application has been increased in the e-commerce or e-business area; the application contains thousands or millions of users, and access control for the users is strongly required. In the previous application, access control module is embedded to the application as a part of the application; it is difficult to modify or enhance the access control module. System developers repeatedly implement the access control module when they begin new application project.

In this paper we intend to implement Java API package named 'Hybrid Access Control (HAC) v1.0' as a library for supporting to build access control module. We consider Java application environment. Because Java is platform-independent and it is able to support wide solutions from mobile (J2ME) to enterprise (J2EE) environment. JSP is used for implementing user interface, Servlet and EJB is used for implementing business logic.

This work was supported by Korea Research Foundation Grant funded by Korea Government(MOEHRD, Basic Research Promotion Fund) (KRF-2005-003-D00366)
[1]Dept. of Computer Science, Dankook Univ.
*Corresponding Author: Sejong Oh(sejongoh@dankook.ac.kr)

Java is a good way to implement enterprise applications.

Java has good security features [7-11]. Security features of Java and researches of Java security are tightly coupled with Java framework. Therefore, application developers are difficult to understand and use Java security features [13-16]. In spite of Java supports function of access control [12], it has some problems. First, Java used property file, it is text format, to store security information. If there is numerous security information data, security manager may mistake to manage them. Second, it is impossible to manage the all data by single manager in the large system; multiple managers share the mission, and it is difficult to control the managers. Third, Java servlet security model use user-role-permission data for access control. If a system developer wants to use different access control model, he should implement the model by himself.

HAC is able to make an offer well-known access control models such as access control list (ACL), mandatory access control (MAC), and role-based access control (RBAC). It also supports new types of access control; it is able to run multiple access control models at the same time according to user's domain or characteristics of information resources. We describe them in section 2. HAC allows configuring access control environment through initialization file. For easy using HAC, we develop security management tool to mange authorization data that is needed for access control.

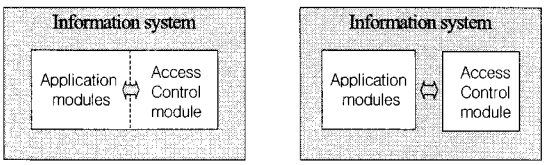
This paper is organized as follows. Section 2 introduces our motivation. Section 3 briefly describes five access control models that HAC library can support. Section 4 describes about contents of HAC library such as Java packages, management tool, and initialization file. Section 5 describes the advantages of HAC library, and the paper is then conclusion in Section 6.

2. Motivation and Related Works

As we mentioned before, most of information systems need access control module, and they implement the module as a part of the systems. In general, access control module is tightly coupled with application modules in the information systems as shown in Fig.1 (a). The tightly coupled systems have the problems:

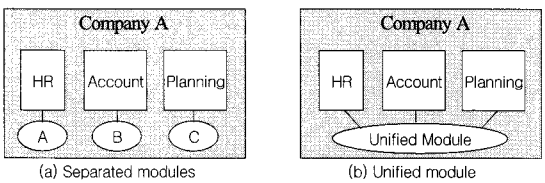
- It is difficult to modify or enhance the access control module. If a developer try to modify access control module, it may lead to change of application modules. Modification of application module also may lead to change of access control module.
- System developers repeatedly implement the access control module when they begin new application project. Tightly coupled module is difficult to reuse.

The solution for above problems is that we make loosely coupled access control module as shown in Fig.1 (b). 'Component' S/W is popular methodology for S/W reuse. In spite of usefulness of loosely coupled access control module, real world requires one more function. Large organization has many departments, and their duty and business style is different from others. It means that each department needs its own access control model. For example, HR department needs MAC, and account department needs RBAC, and planning & management office needs ACL. If we should implement each access control module separately, it may be complicate work as shown in Fig.2 (a). The integrated work of different access control modules is difficult. The solution is that we make unified access control module for multiple access control models as shown in Fig. 2 (b).



(a) Tightly coupled system (b) Loosely coupled system

Fig. 1 Access Control Module in An Information Systems



(a) Separated modules (b) Unified module

(: access control module)

Fig. 2 Different Access Control Modules in An Information System

There is no much works to implement general library for access control. Giuri suggests JRBAC that is improved Java security feature by providing role-based access control mechanism [17]. He extends JDK 1.2 by adding an activation rule. jGuard supports a java API library for web application [18]. Its goal is to provide a security framework based on JAAS. It tries to combine JAAS and RBAC model. In the jGuard, authentications and authorizations are handled by pluggable mechanisms. PERMIS project also aims to support java API library for RBAC model [19]. PERMIS is an infrastructure that provides all the necessary facilities for users to manage privileges and authorization policies and for applications to make authorization decisions. It provides the Attribute Certificate Manager (ACM) and the Bulk Loader for managers to allocate privilege to users.

Those works are useful but cannot meet whole need of real world. Our HAC library is for loosely coupled access control and applying multiple access control models at the same time.

3. Access Control Models

HAC supports five types access control models - ACL, MAC, RBAC, domain-based, and object-centric access control. ACL, MAC, and RBAC are well-known model. Domain-based, and object-centric models are our new suggestion. Last two models are suggested for multi access control environment. The environment requires running several access control models at the same time. In this section, we briefly describe about the five models.

■ ACL, MAC, and RBAC

ACL (access control list) [1-2] is a list of users and groups, with their specific permissions. Unix system uses ACL.

MAC (Mandatory access control)[1,3] means that access control policy decisions are made beyond the control of the individual owner of an object. In the MAC model, users (Subjects) and information objects (Objects) have labels of their security level, and users access is restricted according to their security level.

Role-based access control (RBAC)[3-6] has the central idea of preventing users from accessing company information at their discretion. Instead, access rights are associated with roles, and users are assigned to appropriate roles. The notion of role is an enterprise or organizational concept. Therefore, RBAC allows us to model security from an enterprise perspective, since we can align security modeling to the roles and responsibilities in the company.

■ Domain-based access control

An organization has several departments or business domains. They may need different access control according to their characteristics of business activity. For example, research department needs MAC, and account department needs RBAC, and planning & management office needs ACL. Domain-based access control supports the situation. Each user and system resource belongs to specific domain, and they are dominated by access control model that is assigned to the domain. If a user belongs to domain A, the user cannot access system resources belonging to different domain. For flexibility, domain-based access control allows that user and object can be assigned to multiple domains.

■ Object-centric access control

An organization needs several access control models at the same time. One of principle for selecting access control type for each access event is 'object'. Object means system resource that is target of access. In this model, each object has an 'ac_model (access control model)' property. If a user gives access request, access controller chooses an access control model according to the value of ac_model of the target object. Each object is created during specific business process, and it is accessed for specific business activities. We can say the existence of objects reflects business processes. Remember data modeling is subsequence of requirement analysis for real world. Therefore, it is reasonable to inject ac_model property to objects, and use it as a basis of selecting access control model.

4. Design and Implementation of HAC Library

4.1 Design of HAC Library

In this paper, we design and implement Java library for multiple access control models and management tool for authorization data of the access control models. There is no mean that we just gather independent libraries for multiple access control to single library. We analyze five access control models and elicit common components. As a result, HAC library minimize duplicated component. For example, 'user' and 'information object' are common entities for five access control models. HAC library has only one class for the entities.

Fig.3 shows our implementation scope. HAC Java API library is a core component of our implementation. It contains three packages, and each package includes classes and methods for supporting five types of access control models. Access controller is a part of Java library. We make Java standard API document for HAC. Access control depends on authorization data. System developers can choose file system or database to store authorization data. HAC can connect database by JDBC function. System developers can change parameter values of initialize file to build their own access control environment. Managing authorization data is very complicated work. Therefore, management tool is surely offered with access control library. Our development environment is as follows:

- Operating system : Windows XP
- Java version: JDK 1.5.0
- API documentation tool : javadoc
- IDE : Eclipse SDK 3.5.0
- Database : MySQL 4.0

4.2 Features of HAC Library

HAC library is composed by three packages as shown in Table 1. sec.ac.common package contains classes for common components of five access control models. The classes are related with management of user, object, domain, permission, database connection, user login, and access controller. sec.ac.acl package contains two classes for managing authorization data of ACL model. sec.ac.rbac package contains classes for managing authorization data of RBAC model. The classes are related with management of user-role assignment, permission-role assignment, role-role assignment. Table 2 shows classes of sec.ac.common package. Each class in the three packages has fields or methods Table 3 shows an example of methods of AcUser class in sec.ac.common package. HAC library is distributed as a jar form (ac.jar). Fig.4 is an example of Java source code.

Table 1. Packages of HAC library

Name	Contents
sec.ac.common	classes of common components for five access control models
sec.ac.acl	classes for supporting ACL model
sec.ac.rbac	classes for supporting RBAC model

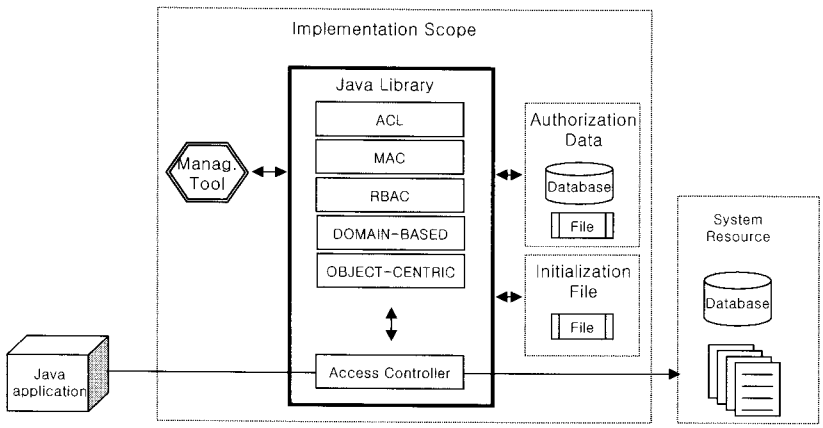


Fig. 3 Implementation Scope

Table 2. Classes of sec.ac.common package

Class Name	Description
AcAccessController	Defines access control methods for five access control models
AcActivation	Contains static methods for managing user activation
AcDomain	Contains static methods for managing domain data
AcObject	Contains static methods for managing object data
AcPerm	Contains static methods for managing permission data
AcUser	Contains static methods for managing user data
AcSchemaActivation	Defines structure of user activation (login) data
AcSchemaDomain	Defines structure of domain data
AcSchemaObject	Defines structure of object (system resource) data
AcSchemaPerm	Defines structure of permission data
AcSchemaUser	Defines structure of user data

Table 3. Methods of AcUser class

Method Name	Description
addObject()	Add new object data
changeDomain()	Change domain of given object
deleteObject()	Delete an object data
deleteObjectDCM()	Delete an object data for domain-based model
getObjectInfo()	Return information of give object name
getObjectList()	Return object list
getRowCount()	Return number of objects
isExistObject()	Check if given object exists or not

AcActivation.java

```
1  package sec.ac.common;
2
3  /* ************************************************************************
4   * CLASS : AcActivation
5   * DESC. : Manage user login and logout information.
6   *
7   * DATE  : 2006/07/01
8   * ************************************************************************/
9
10 import java.text.DateFormat;
11 import java.util.*;
12
13
14 public class AcActivation {
15     static AcCatalog extCatalog = AcCatalog.getInstance();
16
17
18     ////////////////////////////////////////////////////////////////////////
19     public static void addActivation(String user_id) {
20         int result ;
21
22         Date now = new Date();
23         DateFormat fmt = DateFormat.getDateTimeInstance(DateFormat.SHORT, DateFormat.MEDIUM,
24         String active_time = fmt.format(now);
25
26         AcSchemaActivation activ_info_temp = new  AcSchemaActivation() ;
27         activ_info_temp.userID        = user_id ;
28         activ_info_temp.activateTime  = active_time  ;
29
30         extCatalog.activ_info[extCatalog.rowCountActivation] = activ_info_temp ;
31         extCatalog.rowCountActivation++ ;
32     }
```

Fig. 4 Example of Source Code

4.3 Management Tool

All access control model depends on authorization data such as information of user, object, and permission. Managing the authorization data is complex work. Domain-based and object-centric model are more confuse than other models. Therefore, we implement management tool as a bundle S/W of HAC library. The management tool is also implemented by Java, and it fully uses HAC library. The functions of the tool are as follows:

- Edit initialization file
- Insert/update/delete authorization data
- Test user login (activation)
- Test access control
- Monitor user login

Fig. 5 shows an example of management tool. The windows in management tool have different form according to access control model. For example, if system developers choose ACL model for their system, 'Role Management' menu is not shown. System developers can build up different management tool using HAC library for their own purpose.

4.4 Initialization file

We use initialization file to configure access control environment. The initialization file should exist in the Java classpath folder or directory. System developers can edit the initialization file by any text editor or our management tool. Initialization file contains several parameters. Important parameters are as follows:

- Access control model for Java application.
- Selecting method of storing authorization data (FILE or DATABASE)
- System administrator's ID and password
- Folder or directory for authorization data file
- Parameters for database connection

5. Advantages of HAC Library

Using HAC library has much advantages as follows:

- HAC library supports typical types access control models such as ACL, MAC, and RBAC. It also supports special models named domain-based and object-centric access control. The special models widen choice of access control.
- HAC library is very easy to understanding. If a system developer has basic knowledge about access control, he/she may be able to use HAC library without referencing HAC API document.
- System developers do not need to implement access control mechanism themselves. They can choose needed functions from HAC library.

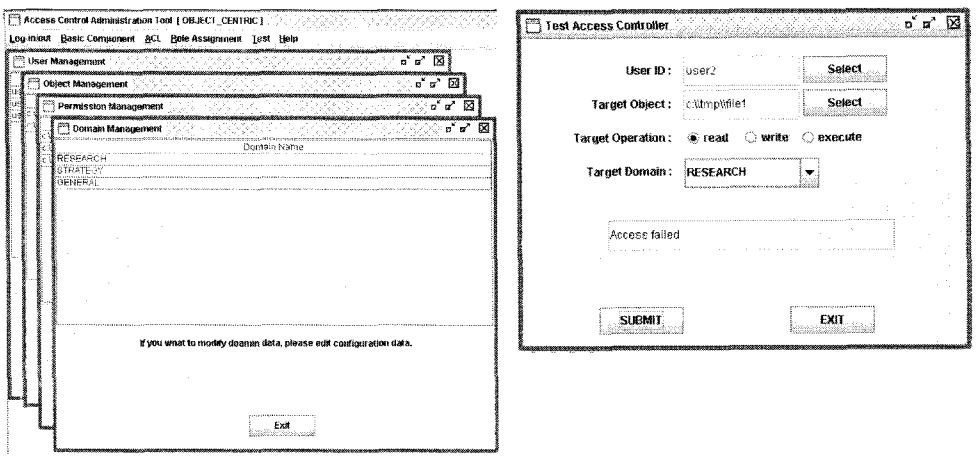


Fig. 5 Management Tool

Table 4. Comparison HAC Library with Other Works

	Pure Java	JRBAC [17]	JGuard [18]	PERMIS [19]	HAC Library
ACL	○		○	○	○
MAC					○
RBAC	△	○	○	○	○
Domain-based					○
Object-centric					○
Support API library	○		○	○	○
Support admin. tool					○

- It is easy to setup customized access control environment through editing initialization file. It has feature of general test property file.
- Bundle management tool helps easy control of authorization data. System developers may implement their own management tool using HAC library.
- It is useful for education of access control. Students can compare several access control models. They can test the behavior of each access control mechanism.
- HAC library may continuously enhanced, and system developers who are using HAC library can trust the access control is safe.

Table 4 shows comparison HAC library with other works.

6. Conclusion

Many organizations or information systems need access control mechanism. In times past, system developers should implement the access control mechanism by themselves. In this paper, we implement HAC library to support access control. HAC library supports five types access control models with management tool. Next version of HAC library may contain the feature of Java web application. Web application is a popular trend of building information system. HAC library reduces the effort that system developers implement access control mechanism by themselves. It also supports secure access control for information system that requires multiple access control models at the same time.

References

[1] C.P. Pfleger, Security in Computing, Prentice-Hall International Inc, 2nd Ed., pp 244-250, 1997.

[2] D Russel, G.T.Sr. Gangemi, Computer Security Basics, O'Reilly & Associates, Inc., pp67-77, 1991.

[3] E.G. Amoroso, Fundamental of Computer Security Technology, PTR Prentice Hall, pp101-112, 1994.

[4] R. Sandhu, E.J.Coyne, H.L. Feinstein, C.E. Youman, "Role-Based Access Control Method", IEEE Computer, vol.29, pp38-47, 1996.

[5] D. Ferraio, J. Cugini, R. Kuhn, "Role-based Access Control (RBAC): Features and motivations", Proc. of 11th Annual Computer Security Application Conference, 1995.

[6] R. Sandhu, "Rationale for the RBAC96 Family of Access Control Models", Proc. of ACM Workshop on Role-Based Access Control, 1995.

[7] J. Garms, D. Somerfield, Professional Java security, Wrox press, 2002.

[8] Core Java Security and the Java Platform, http://java.sun.com/security/

[9] Java Cryptography Extension (JCE), http://java.sun.com/products/jce/

[10] Java Secure Socket Extension (JSSE), http://java.sun.com/products/jsse/

[11] JavaTM Virtual Machines, http://java.sun.com/j2se/1.5.0/docs/guide/vm/ index.html

[12] Java Authentication and Authorization Service (JAAS), http://java.sun. com/products/jaas/

[13] D.S. Wallach, D. Balfanz, D. Dean, E. W.Felten, "Extensible security architectures for Java", ACM SIGOPS Operating Systems Review, Vol.31 Issue 5, 1997.

[14] R. Barbuti, C. Bernardeschi, N. De Francesco, "Computer security: Checking security of Java bytecode by abstract interpretation", Proc. of the 2002 ACM

symposium on Applied computing, 2002.

[15] L. Koved, M. Pistoia, A. Kershenbaum, "Access rights analysis for Java", ACM SIGPLAN Notices, Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, Vol. 37 Issue 11, 2002.

[16] M. Hauswirth, C. Kerer, R. Kurmanowytsch, "A secure execution framework for Java", Proc. of the 7th ACM conference on Computer and communications security, 2000.

[17] L. Giuri, "Role-based Access Control in Java", Proc. of the 3th ACM workshop on Role-Based Access Control, 1998.

[18] jGuard Project, http://sourceforge.net/projects/jguard

[19] PERMIS Project, http://sec.isi.salford.ac.uk/permis/

Se-Jong Oh [Regular Member]

- Feb, 1989 : Sogang Univ, dept. of computer science (BS)
- Feb, 1991 : Sogang Univ, dept. of computer science (MS)
- Aug, 2001 : Sogang Univ, dept. of computer science (Ph.D)
- Sep. 2003 ~ : Assistant Professor, dept. of computer science, Dankook Univ.

<research area>
access control for enterprise and distributed systems, ERP, secure DBMS, and ubiquitous security