Secure Information Sharing Using Attribute Certificates and Role Based Access Control*

Ganesh Godavari, and Edward Chow
Department of Computer Science, University of Colorado, 1420 Austin Bluffs Parkway
Colorado Springs, Colorado 80917 USA
gkgodava@cs.uccs.edu, chow@cs.uccs.edu

Abstract

World Wide Web (WWW) has become a part of the way in which we transfer information. Security and authorization have become a concern for sharing of critical information. Role based access control (RBAC) provides some flexibility to security management. Public key infrastructure (PKI) can provide a strong authentication. Privilege management infrastructure (PMI) as a new technology can provide strong authorization. In order to satisfy mentioned security requirements, we have established a role based access control infrastructure and developed a prototype that uses X.509 public key certificates (PKCs) and attribute certificates (ACs). In this paper we explore the use of Attribute Certificates with RBAC for supporting large scale secure information sharing. We use Ldap servers for storing ACs, PKC to provide authentication and authorization services for Web services. The scheme proposed has many advantages that satisfy the needs of inter-organization information sharing using attribute certification

1 Introduction

As the WWW is fast becoming a place for sharing of information, piracy, and misuse of information are fast becoming a threat. Security and Authorization are becoming necessary. This situation not only provides excellent business opportunities but also research challenges. One of the most challenging problems is managing large information sharing systems is the complexity of security administration, particularly access control using Role Based Access Control simplifies access control administration and provides better manageability in enterprise environments by allowing permissions to be managed in terms of user job roles [9]. RBAC maps user job roles to application permissions so that access control administration can be accomplished in terms of users job role. This means that administrators will have to set up roles, such as employee, manager and administrator, with out having to change the access permission on each object.

Many of the e-commerce applications require authentication services, in addition to the basic services provided by Public Key Infrastructures (PKI), to allow users to do what they are allowed to do. Authentication means that that the sender of a message or transaction is verified to be who they claim to be, while Authorization means that someone who has the authority to do, so he/she can initiate or progress a transaction, process or activity. In simple terms, Authentication is what is required to gain access eg, a passport, driving license or in computing terms password, strong authentication. Authorization is details of where you are permitted to go, once you are authenticated. Public key certificate (PKC) strongly binds a public key to its subject (country, location, organization unit etc.) helping to identify the holder of the certificate. Attribute certificates have been proposed as a solution for the authorization services. The Attribute certificates are designed to convey (potentially short-lived) attributes about a given subject to facilitate flexible and scalable privilege management. The attribute certificate may point to a public-key certificate that can be used to authenticate the identity of the attribute certificate holder.

Some research and development efforts have been done in this area [2, 15, 7], but these efforts are still in primary phase, and no authorization mechanism

*Draft Report 0.1

is widely accepted. We were motivated by the need of using PKI, PMI and RBAC concepts to construct an authorization mechanism which uses the PERMIS [1, 2] model of storing the user's roles in ACs. Access control decisions are driven by an authorization policy, and the authorization policy is also stored in an AC.

The organization of this paper is as follows. Section 2 provides an overview of the related research. Section 3 describes the architecture of the software system implemented. In Section 4 we analyze the results. Finally, in Section 5, we discuss future directions for research.

2 Related Research Technologies

2.1 Role Based Access Control

Role-based access control [5, 14, 12, 16, 3] has gained attention as a proven alternative to traditional discretionary (DAC) and mandatory access control (MAC) mechanisms. RBAC helps to specify organization's security policies reflecting its organizational structure. In the core RBAC, a user can be assigned one or more roles, and a role can be assigned to one or more users. roles are based on the user's job responsibilities in the organization. This provides for flexibility and finer granularity during assignment of access permissions to roles and users to roles. In role-based model the role hierarchy partially determines which roles and permissions are available to users via various inheritance . for example A senior role can inherit permissions from junior roles. A user establishes a session during which he activates some subset of roles that he is a member of. RBAC provides Static Seperation of Duty (SSD) relations to prevent conflict of interests that arise when a user gains permissions associated with conflicting roles, and Dynamic Seperation of Duty relations to place constraints on roles that can be activated in a users session.

The RBAC in Figure 1 consists of 1) a set of users (USERS) where a user is an intelligent autonomous agent, 2) a set of roles (ROLES) where a role is a job function, 3) a set of objects (OBS) where an object is an entity that contains or receives information, 4) a set of operations (OPS) where an operation is an executable image of a program, and 5) a set of permissions (PRMS) where a permission is an approval to perform an operation on objects. The cardinalities of the relationships are indicated by the absence (denoting one) or presence of arrows (denoting many) on the corresponding associations. For example, the association of user to session is one-to-many. All other associations shown in the figure are many-to-many. The association labeled Role Hierarchy defines the inheritance relationship among roles.

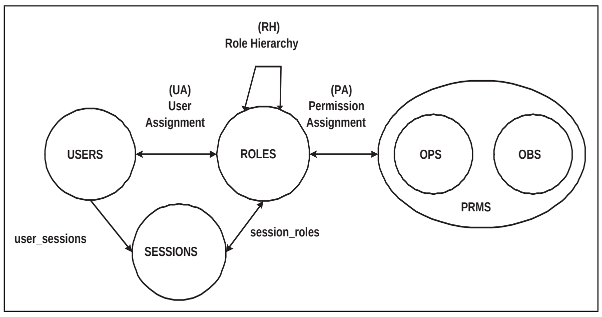


Figure 1: RBAC model

Further more information about RBAC is available at [9].

2.2 Privilege Management Infrastructure

PMI is the information security infrastructure that assigns privilege attribute information such as privilege, capability, and role, etc., to users and issues and manages it using the X.509 Attribute Certificate. The PMI supports access control service using the user's privilege management in application services. The function of the PMI is to specify the policy for the attribute certificate issuance and management. Then, the PMI carries out the AC-related management functions such as issuing, updating, and revoking an attribute certificate based on a specified policy.

In PMI the ACs issuer is called Attribute Authority (AA). ACs are digitally signed by the AA, so they are tamper-resistant. The trusted root is called source of authority (SOA). When a user's authorization permissions need to be revoked, AA will issue an attribute certificate revocation list (ACRL) containing the list of ACs no long to be trusted. There are two primary models for distribution of attribute certificates: the 'push' or 'pull' model. The 'push' model is suitable when the client's permissions should be authenticated/validated in the client's 'home' domain, whereas the 'pull' model is suitable when the client's privelages should be authenticated in the inter-domain.

Further more information about AC is available at [13]

3 Implementation

3.1 System Overview

Our access control system is designed to support RBAC using X.509 PKIs and ACs. The authentication is implemented by PKI, and the authorization is implemented by AC. Role information is stored in User Role Specification AC's (see section 'Administration tool'). All the access control decisions are made based on authorization policies. They are written in XML and stored in ACs. ACs and their corresponding PKIs are all stored in LDAP servers [10]. In our current prototype implementation we have a simple policy specification file. (we need to evaluate policy specification using a) eXtensible Access Control Markup Language (SAML) [4] b) PERMIS X.500 PMI RBAC policy [1]).

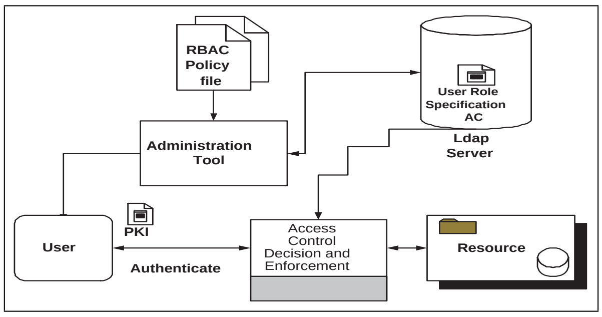


Figure 2: Overview of the SIS Access Control System

Our current RBAC policy specification file is shown below

<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!-======= SIS rbac example =======->
<sis>
<Role>Manager</Role >
<Group>Info Share</Group >
<OU>UCCS</OU >
</sis>

- *RBAC Policy file:* specify the roles and what privileges the role can have on the resources. Access control decision are made based on these privileges.
- *Administration Tool:* is used for creating key pair, PKIs, User Role Specification ACs.
- *Ldap Server:* stores the user's information along with User Role Specification ACs and Delegated Role Specification AC's.
- *Access Control Decision and Enforcement:* executes the function of authorization and informs the target if the user has the privilages or not.
- *Resources:* they may be web servers, database servers, or any other format of resources.

3.2 Administration tool

OpenSSL [11] provides strong open source cryptographic library for X509, and SSL&TLS. Currently OpenSSL does not have any support for Attribute Certificates except for RFC 3281 AC's ASN.1 object definitions. Attribute Certificates generation code was written using the ASN.1 object definitions in OpenSSL. There are two types of Attribute Certificates in our proposed architecture system:

1. 'User Role specification' attribute certificate which tells what privilege(s) a user has. It is used by the decision making service to make a decision to determine whether a user has access information or resources available in application services.
2. 'Delegated Role Specification' attribute certificate which tells what privilages are given for a resource(s) by a user of higher authority.

In our prototype we adopt AC 'Pull' model, so the role ACs are not given to users. The 'User Role specification' and 'Delegated Role Specification' ACs are all stored in LDAP servers. We currently donot have support for ACRL (Attribute Certificate Revocation List) needed by the 'push model'. We plan to provide this capability soon. This handicap can be circumvented by deleting the entry from the LdapServer.

3.3 Access Control Decision and Enforcement

The access control engine is implemented as an Apache [6] module. It is responsible for authorization service for web requests between user and the requested target file(s). This framework seperates Authentication service, provided by ModSSL [8] from authorization service. Our prototype consists of following components: Initiator (e.g. a browser), Target (web file request), Access Control Decision and Enforcement (ACDE) provided by the apache_sis_module.

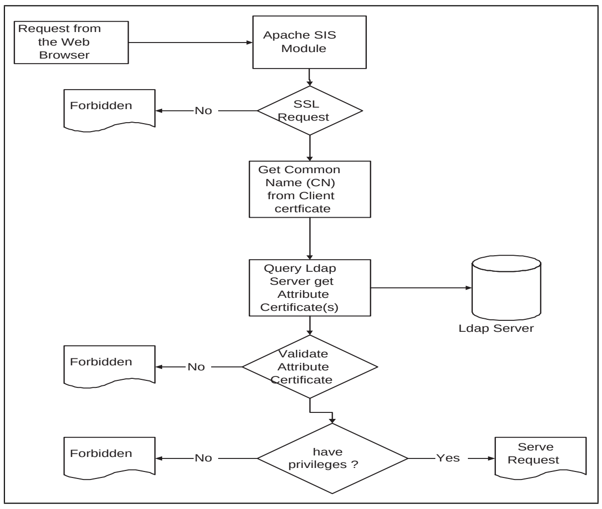

Request from the Web Browser
Apache SIS Module
SSL Request
No
Forbidden
Get Common Name (CN) from Client certficate
Query Ldap Server get Attribute Certificate(s)
Ldap Server
Validate Attribute Certificate
No
Forbidden
have privileges ?
No
Forbidden
Yes
Serve Request

Figure 3: Control Flow in Access Control Engine

The initiator submits web access request to the target, The ACDE gets the initiators details from the submitted public key and queries the LDAP server for validation. Once the user is found ACDE retrieves the User's Attribute Certficates and validates his/her User Role specification AC. Once the user is validated it makes a decision whether the user has the required privilage to access the information. Only users with Valid privilages are allowed to have access to the target by the ACDE.

References

[1] Chadwick, D. W., and Otenko, A. Rbac policies in xml for x.509 based privilege management. In *Int. Conf. On Information Security* (2002), pp. 39–53.

[2] Chadwick, D. W., and Otenko, A. The permis x.509 role based privilege management infrastructure. *Future Gener. Comput. Syst. 19*, 2 (2003), 277–289.

[3] D. Ferraiolo, J. C., and Kuhn, D. R. Role-based access control: Features and motivations. *Computer Security Applications Conference* (1995), 241.248.

[4] eXtensible Access Control Markup Language (XACML). authorization policies specification in xml. *http://sunxacml.sourceforge.net/* (2004).

[5] Ferraiolo, D., and Kuhn., D. R. Role-based access control. *15th NIST-NCSC National Computer Security Conference* (1992), 554–563.

[6] Jakarta. http://www.apache.org/. *Apache Software Foundation* (2004).

[7] Javier Lopez, Antonio Mana, J. J. O. J. M. T., and Yague, M. I. Integrating pmi services in corba applications. *Comput. Stand. Interfaces 25*, 4 (2003), 391–409.

[8] ModSSL. http://www.modssl.org/. *Apache Interface to OpenSSL* (2004).

[9] NIST. Role-based access control. *http://csrc.nist.gov/rbac/* (2004).

[10] OpenLDAP. The open source lightweight directory access protocol (ldap). *http://www.openldap.org/* (2004).

[11] OpenSSL. The open source toolkit for ssl/tls. *http://www.openssl.org/* (2004).

[12] R. Sandhu, E. J. Coyne, H. L. F., and Youman., C. E. Role-based access control models. *IEEE Computer 29* (1996), 38–47.

[13] S. Farrell, R. H. An internet attribute certificate profile for authorization. *http://www.ietf.org/rfc/rfc3281.txt* (2002).

[14] Sterne., D. F. A tcb subset for integrity and role-based access control. *15th NIST-NCSC National Computer Security Conference NIST/NSA* (1992).

[15] Thompson, M. R., Essiari, A., and Mudumbai, S. Certificate-based authorization policy in a pki environment. *ACM Trans. Inf. Syst. Secur. 6*, 4 (2003), 566–588.

[16] von Solms, S. H., and van der Merwe, I. The management of computer security profiles using a role-oriented approach. *Comput. Secur. 13*, 9 (1994), 673–680.

4 Appendix

4.1 Installation

Currently SIS 0.1 has been tested on x86 linux. It currently uses OpenSSL 0.9.7d for cryptographic functions, Apache 1.3.31 for web services, and ModSSL 2.8.18-1.3.31 for providing SSL/TLS support in Apache. The following instructions install SIS 0.1 as well as OpenSSL 0.9.7d, Apache 1.3.31, and ModSSL 2.8.18-1.3.31.

1. download the source code of SIS 0.1 from http://blanca.uccs.edu/ infoshare/sis-0.1.tar.gz

2. download the distributions of Apache, mod_ssl and OpenSSL
   ```
   $ lynx http://httpd.apache.org/dist
   /httpd/apache_1.3.31.tar.gz
   $ lynx ftp://ftp.modssl.org/source/mod_ssl-
   2.8.18-1.3.31.tar.gz
   $ lynx ftp://ftp.openssl.org/source/openssl-
   0.9.7d.tar.gz
   ```

3. extract the distributions of Apache, mod_ssl and OpenSSL
   ```
   $ tar -xvzf apache_1.3.31.tar.gz
   $ tar -xvzf mod_ssl-2.8.18-1.3.31.tar.gz
   $ tar -xvzf openssl-0.9.7d.tar.gz
   ```

4. Build OpenSSL
   ```
   $ cd openssl-0.9.7d
   $ ./config
   $ make
   $ cd ..
   ```

5. extract SIS module into apache
   ```
   $ tar -xvzf sis-0.1.tar.gz
   $ mv sis-0.1 apache_1.3.31/src/modules/
   $ cd ..
   ```

6. Build and install the SSL and SIS aware Apache
   ```
   $ cd mod_ssl-2.8.18-1.3.31
   $./configure
   –with-apache=../apache_1.3.31
   –with-ssl=../openssl-0.9.7d
   –prefix=/usr/local/apache
   –activate-module=src/modules/sis-
   0.1/mod_auth_ldap.c
   $ cd ..
   $ cd apache_1.3.31
   $ make
   $ make certificate
   $ make install
   ```

Now SIS 0.1 module has been created and added to apache

Put a section like below in the httpd.conf file:

```
<Directory ”/usr/local/apache/htdocs/foo”>
Options Indexes FollowSymLinks
AllowOverride All
LDAP_Server ncxdcr2x1.uccs.edu
LDAP_Port 389
Base_DN ”o=University of Colorado at Col-
orado Springs”
AuthLDAPAuthoritative on
Bind_DN ”cn=University of Colorado at Col-
orado Springs Admin,o=University of Colorado
at Colorado Springs”
#Bind_Pass ”secret”
UID_Attr cn
CACertificateFile /home/gkgodava/rbac/3281/openssl/opens
0.9.7c/3281/openssl-ac-supp/ca-bundle.crt
SISrequire ”sis:1.Role:1.=Manager,sis:1.Group:1.=Info
Share,sis:1.OU:1.=UCCS,”
</Directory>
```

where

- *AuthLDAPAuthoritative* Setting this directive to 'no' (by default it is 'yes') allows for both RBAC authorization to be performed using AC's

- *LDAP_Server* The hostname of your LDAP server, e.g. ldap.foo.com. If this directive is not defined in the config file for a directory, then the control will be given back so that you can authenticate with other mechanism.
- *LDAP_Port* The port on LDAP server. The default and standard port number for LDAP is 389.

- *Base_DN* The LDAP Base Distinguished Name (DN) for search.
- *Bind_DN* If your LDAP server does not allow anonymous binding (e.g. MS Windows 2000 Active Directory), specify the full Distinguised Name (DN) to bind to the server.
- *Bind_Pass* The bind password (in plain text).
- *UID_Attr* The attribute to use in LDAP search. The default LDAP attribute is 'cn'.
- *CACertificateFile* The path for the Attribute Authrority certificate.
- *SISrequire* ACL role specification (see section 4.2 for further details).

DO NOT forget to edit the above section and make sure to change the LDAP_Server to your Ldap server, Base_DN and other required attribute as well.

4.2 Rbac polify ACL specification

The Access Control List rule specification scheme should be flexible enough to account for exact tag name or rule field indicated in the rule specification. Below shows an example that illustrates this point.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!-======= SIS rbac parsing example =======->
<purchase>
<customerName>CCL</customerName>
<customerID>111222333</customerID>
<item>
<productID>309121544</productID>
<unitPrice>5000</unitPrice>
<subTotal>50000</subTotal>
</item>
<item>
<productID>309121538</productID>
<unitPrice>200</unitPrice>
<subTotal>2000</subTotal>
</item>
<totalAmount>52000</totalAmount>
</purchase>
<purchase>
<customerName>CDL</customerName>
<customerID>111222444</customerID>
<item>
<productID>30913555</productID>
<unitPrice>3000</unitPrice>
<subTotal>20000</subTotal>
</item>
<totalAmount>20000</totalAmount>
</purchase>
```

In the XML document shown above, some of the tags are repeated, e.g., purchase, item, totalAmount. Hence, a rule syntax is needed to allow for selecting a particular set of tags in the rule set. Here is an example of a scheme that addresses this problem. To specify a rule based on subTotal value present in the second item tag within the first purchase tag, the rule will be specified as 'purchase:1.item:2.subtotal'. As another example, 'purchase:2.totalAmount' specifies a rule based on the totalAmount tag present within the second purchase tag.

5 Administration Tool

Administration tool is used to simplify the process of creating, managing PKC and AC certificates. it uses OpenSSL 0.9.7d for generation of certificates

1. download the source code of Admin tool from http://blanca.uccs.edu/ infoshare/sis-admin-0.1.tar.gz

2. extract SIS Admin tool
   $ tar -xvzf sis-admin-0.1.tar.gz
   $ Make

3. create CA certificate using the command (make sure to back up the cacert.pem and cakey.pem)
   $ sh admin.sh ca

4. create User certificates using the command
   $ sh admin.sh pkc <username>
   certificates and key are created with the username e.g. <username>-cert.pem (certificate); <username>-key (key).
   also browser<username>.p12 will be created this can be used for storing the certificate in the web browser of the user.

5. create attribute certificates using the command
   $ sh admin.sh ac <issuer-certficate><Rbac-policy-file ><outputfilename >
   creates both 'User Role specification' and 'Delegated Role Specification' AC depending on the issuer.

6 Ldap Server

Ldap server is used to store PKC and AC certificates generated using the Administration tool. OpenLdap server cannot store attribute certficates. Follow the below instructions on how to store attribute certificates in OpenLdap server.

1. download the patch for open-ldap-2.0.27 from http://blanca.uccs.edu/ infoshare/patchfileldap

2. apply the patch to add attribute certificate schema
   $ patch -p0 <patchfileldap

3. configure slapd.conf. below is a snippet of the configuration file
   ***************Snippet Of slapd.conf **********************

   database ldbm
   suffix "o=UCCS"
   rootdn "cn=UCCS Admin,o=UCCS"
   rootpw secret
   index cn,sn,st pres,eq,sub

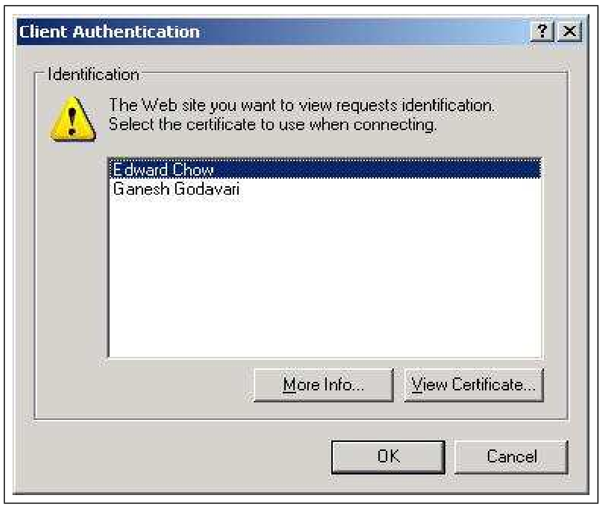
4. start ldap server
   service ldap start

5. add an entry to the ldap server
   sample ldif file can be downloaded from http://blanca.uccs.edu/ infoshare/sample.lidf

we are not able to hold the AC as an object in Ldap server. so make sure you can specify the whole path of the AC's location.

7 Demo

Start the Apache webserver using '/usr/local/apache/bin/apachectl startssl'
Install the browser certificates in your favorite browser.
connect to the webserver (https://<machinename>///<directory specified in apache configuration >/filename.html
when prompted for certificate submit the generated the installed certificate to view files.

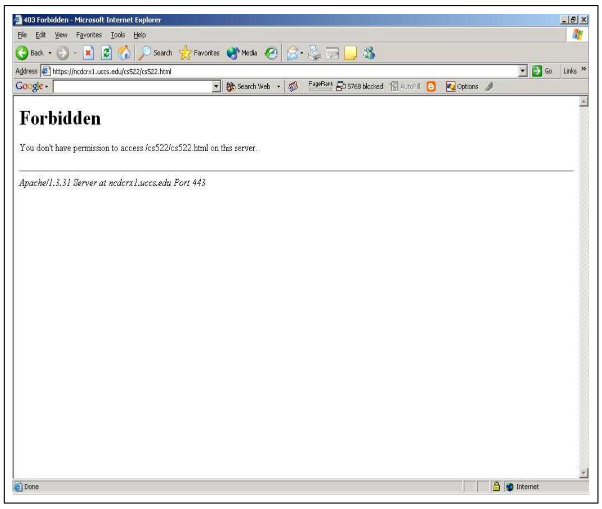

Client Authentication

Identification

The Web site you want to view requests identification.
Select the certificate to use when connecting.

Edward Chow
Ganesh Godavari

Figure 4: Certificate prompt


403 Forbidden - Microsoft Internet Explorer

Address https://ncdcrx1.uccs.edu/cs522/cs522.html

Forbidden

You don't have permission to access /cs522/cs522.html on this server.

Apache/1.3.31 Server at ncdcrx1.uccs.edu Port 443

Figure 5: access of the file without right privilages