Role-Based Privilege Management Using Attribute Certificates and Delegation

Gail-Joon Ahn, Dongwan Shin, and Longhua Zhang

University of North Carolina at Charlotte, Charlotte, NC 28232, USA
{gahn,doshin,lozhang}@uncc.edu

Abstract. The Internet provides tremendous connectivity and immense information sharing capability which the organizations can use for their competitive advantage. However, we still observe security challenges in Internet-based applications that demand a unified mechanism for both managing the authentication of users across enterprises and implementing business rules for determining user access to enterprise applications and their resources. These business rules are utilized for privilege management or authorization in a security context. In this paper, we design a role-based privilege management leveraging access control models and X.509 attribute certificate. We attempt to develop an easy-to-use, flexible, and interoperable authorization mechanism. Also, we demonstrate the feasibility of our architecture by providing the proof-of-concept prototype implementation using commercial off-the-shelf technologies.

1 Introduction

Many organizations have transited from their old and disparate business models based on ink and paper to a new, consolidated ones based on digital information on the Internet. The Internet is uniquely and strategically positioned to address the needs of a growing segment of population in a very cost-effective way. It provides tremendous connectivity and immense information sharing capability which the organizations can use for their competitive advantage. However, we still observe security challenges in Internet-based applications that demand a unified mechanism for both managing the authentication of users across enterprises and implementing business rules for determining user access to enterprise applications and their resources. These business rules are utilized for privilege management or authorization in a security context [13]. In this paper, we often use the term authorization and access control as an identical notion of privilege management. Authentication mechanisms have been practiced at considerable length and various authentication schemes such as SSL, LDAP-based, or secure cookies-based have been widely accepted. Unlike authentication mechanisms, authorization mechanisms which can conveniently enforce various business rules from different authorization domains among various applications still need to be investigated.

Role-based access control (RBAC) has been acclaimed and proven to be a simple, flexible, and convenient way of managing access control [6, 15]. This extremely simplifies management of privileges, reducing complexity and potential

S. Katsikas, J. Lopez, and G. Pernul (Eds.): TrustBus 2004, LNCS 3184, pp. 100–109, 2004.

errors in directly assigning privileges to users. Another issue is to support such a simplified privilege management among distributed Internet-based enterprise applications. Privilege management infrastructure (PMI) [4, 5] has recently been introduced allowing us to establish the trustworthiness among different authorization domains as long as each of them keeps the meaning of attributes intact.

Our objective in this paper is to design a role-based privilege management leveraging RBAC features and X.509 attribute certificate in PMI. We attempt to develop an easy-to-use, flexible, and interoperable authorization mechanism. We also seek to address the issue of how to advocate selective information sharing in internet-based enterprise applications while minimizing the risks of unauthorized access.

The rest of this paper is organized as follows. Section 2 shows previous researches related to our work. Section 3gives an overview of background technologies. Section 4 describes our approach to designing a role-based privilege management with attribute certificates and delegation including system architecture and authorization policies. Implementation details are described in Section 5. Section 6 discusses lessons learned from our experiment and concludes the paper.

2 Related Works

Several researchers have been trying to accommodate RBAC features into large-scale systems of intranet or extranet focusing on various applications such as database systems, web servers, or web-based workflow systems. At the same time, delegation has been studied by a number of researchers as an important factor for secure distributed computing environment [7].

In the OSF/DCE environment [11], privilege attribute certificate (PAC) that a client can present to an application server for authorization was introduced. PAC provided by a DCE security server contains the principal and associated attribute lists, which are group memberships. This approach focused on the traditional group-based access control.

Similarly, Thompson et al. [18] developed a certificate-based authorization system called Akenti for managing widely distributed resources. It was especially designed for system environments where resources have multiple stakeholders and each stakeholder wants to impose conditions for access. Their approach emphasized the policy-based access control in a distributed environment.

Also, several studies have been carried out to make use of RBAC features with the help of public-key certificates [1, 12]. Public-key certificates were used to contain attribute information such as role in their extension field. To add role information into public key certificates, however, may cause problems such as shortening of certificates' lifetime and complexity of their management [17].

In general, delegation is referred to as one active entity in a system delegates its authority to another entity to carry out some functions. In role-based systems, the delegated authorities are roles. The requirements related to role-based delegation have been identified in the literature [2, 8, 21]. A work closely related

to ours is RBDM0 model proposed by Barka and Sandhu [2]. They developed a simple role-based delegation model. They explored some issues including revocation, delegation with hierarchical roles, partial delegation, and multi-step delegation. One limitation of RBDM0 is that this work does not address the relationships among each component of a delegation, which is a critical notion to the delegation model. A number of researchers have looked at the semantics of authorization, delegation, and revocation. Li et al. proposed a logic for authorizing delegation in large-scale, open, distributed systems [3, 10]. But in their logic, role-based concepts were not fully adopted; neither did they address revocation adequately.

3 Background Technologies

3.1 Role-Based Access Control

RBAC is an alternative policy to traditional mandatory access control (MAC) and discretionary access control (DAC). As MAC is used in the classical defense arena, the policy of access is based on the classification of objects such as top-secret level [14]. The main idea of DAC is that the owner of an object has discretionary authority over who else can access that object [9]. But RBAC policy is based on the role of the subjects and can specify security policy in a way that maps to an organization's structure. A general family of RBAC models called RBAC96 was defined by Sandhu et al [15]. Motivation and discussion about various design decisions made in developing this family of models is given in [15, 16]. Also, there are variations regarding distributed systems [20].

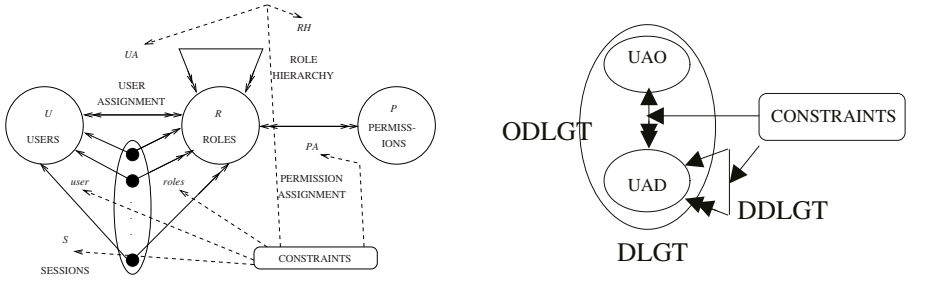
Figure 1(a) shows (regular) roles and permissions that regulate access to data and resources. Intuitively, a user is a human being or an autonomous agent, a role is a job function or job title within the organization with some associated semantics regarding the authority and responsibility conferred on a member of the role, and a permission is an approval of a particular mode of access to one or more objects in the system or some privilege to carry out specified actions. Roles are organized in a partial order $\geq$, so that if $x \geq y$ then role x inherits the permissions of role y. Members of x are also implicitly members of y. In such cases, we say x is senior to y. Each session relates one user to possibly many roles. The idea is that a user establishes a session and activates some subset of roles that he or she is a member of (directly or indirectly by means of the role hierarchy). A user may have multiple sessions open at the same time, each in a different window on the workstation screen for instance. Each session may have a different combination of active roles. The concept of a session equates to the traditional notation of a subject in access control. A subject is a unit of access control, and a user may have multiple subjects (or sessions) with different permissions active at the same time.

3.2 Privilege Management Infrastructure

PMI is based on the ITU-T Recommendation of directory systems specification [4], which introduced PKI in its earlier version. Public-key certificates are used

in PKI while attribute certificates are a central notion of PMI. Public-key certificates are signed and issued by certification authority (CA), while attribute certificates are signed and issued by attribute authority (AA). PMI is to develop an infrastructure for access control management based on attribute certificate framework. Attribute certificates bind attributes to an entity. The types of attributes that can be bound are role, group, clearance, audit identity, and so on. Attribute certificates have a separate structure from that of public key certificates.

PMI consists of four models: general model, control model, delegation model, and roles model. General and control models are required, whereas roles and delegation models are optional. The general model provides the basic entities which recur in other models.



(a) RBAC Model (b) Delegation Relation

Fig. 1. RBAC and Delegation.

4 Role-Based Privilege Management

4.1 Adopting Attribute Certificate

Our approach is based on basic entities in PMI. It consists of three foundation entities: the object, the privilege asserter, and the privilege verifier. The control model explains how access control is managed when privilege asserters request services on object. When the privilege asserter requests services by presenting his/her privileges, the privilege verifier makes access control decisions based upon the privilege presented, privilege policies, environmental variables, and object methods. PMI roles model also introduces two additional components: role assignment and role specification. Role assignment is to associate privilege asserters with roles, and its binding information is contained in attribute certificate called role assignment attribute certificate. The latter is to associate roles with privileges, and it can be contained in attribute certificate called role specification attribute certificate or locally configured at a privilege verifier's system. Our approach is based upon PMI roles model. Accordingly, two different attribute certificates are employed: role assignment attribute certificate ($RAAC$) and role specification attribute certificate ($RSAC$). The integrity of the bindings is guaranteed through digital signature in attribute certificate.

4.2 Constrained Role-Based Delegation

Zhang et al. [21] introduced RDM2000 (role delegation model 2000) for user-to-user delegation in role-based systems. Our work is based on RDM2000. It formalizes the relationship between two user assignments that form a delegation relation (DLGT), as shown in Figure 1(b). We first define a new relation called delegation relation (DLGT). It includes sets of three elements: original user assignments UAO, delegated user assignment UAD, and constraints. The motivation behind this relation is to address the relationships among different components involved in a delegation. In a user-to-user delegation, there are four components: a delegating user, a delegating role, a delegated user, and a delegated role. A delegation relation is one-to-many relationship on user assignments. It consists of original user delegation (ODLGT) and delegated user delegation (DDLGT). We assume each delegation relation may have a duration constraint associated with it. If the duration is not explicitly specified, we consider the delegation as permanent unless another user revokes it. The function *Duration* returns the assigned duration-restriction constraint of a delegated user assignment. If there is no assigned duration, it returns a maximum value. Our delegation model has the following components and theses components are formalized from the above discussions.

- T is a set of duration-restricted constraint.
- DLGT $\subseteq$ UA $\times$ UA is one to many delegation relation. A delegation relation can be represented by $(u,\ r,\ u',\ r') \in$ DLGT, which means the delegating user u with role r delegated role r' to user u'.
- ODLGT $\subseteq$ UAO $\times$ UAD is an original user delegation relation.
- DDLGT $\subseteq$ UAD $\times$ UAD is a delegated user delegation relation.
- DLGT = ODLGT $\cup$ DDLGT.

In some cases, we may need to define whether or not each delegation can be further delegated and for how many times, or up to the maximum delegation depth. We introduce two types of delegation: single-step delegation and multi-step delegation. Single-step delegation does not allow the delegated role to be further delegated; multi-step delegation allows multiple delegations until it reaches the maximum delegation depth. The maximum delegation depth is a natural number defined to impose restriction on the delegation. Single-step delegation is a special case of multi-step delegation with maximum delegation depth equal to one.

Also, we have an additional concept, delegation path (DP) that is an ordered list of user assignment relations generated through multi-step delegation. A delegation path always starts from an original user assignment. We use the following notation to represent a delegation path.

$uao_0 \rightarrow uad_1 \rightarrow uad_i \rightarrow uad_n$

Delegation paths starting with the same original user assignment can further construct a delegation tree. A delegation tree (DT) expresses the delegation paths in a hierarchical structure. Each node in the tree refers to a user assignment and each edge to a delegation relation. The layer of a user assignment in the tree

is referred as the delegation depth. The function Prior maps one delegated user assignment to the delegating user assignment; function Path returns the path of a delegated user assignment; and function Depth returns the depth of the delegation path.

Constraints are an important aspect of RBAC and can lay out higher-level organizational policies. In theory, the effects of constraints can be achieved by establishing procedures and sedulous actions of security administrators [6]. Constraints are enforced by a set of integrity rules that provide management and regulators with the confidence that critical security policies are uniformly and consistently enforced. In the framework, when a user delegates a role, all context constraints that are assigned to the user and anchored to the delegated role are delegated as well.

Rule-Based Policy Specification Language. We also define policies that allow regular users to delegate their roles. It also specifies the policies regarding which delegated roles can be revoked. A rule-based language is adopted to specify and enforce these policies. It is a declarative language in which binds logic with rules. The advantage is that it is entirely declarative so it is easier for security administrator to define policies.

A *rule* takes the form:
$H \leftarrow F1\&F2\& \ldots \&Fn$
where H, F1, F2,. . . , Fn are Boolean functions.

There are three sets of rules in the framework: basic authorization rules specify organizational delegation and revocation policies; authorization derivation rules enforce these policies in collaborative information systems; and integrity rules specify and enforce role-based constraints. For example, a user-user delegation authorization rule forms as follows:
$can\_delegate(r, cr, n) \leftarrow .$

where r, cr, and n are elements of roles, prerequisite conditions, and maximum delegation depths respectively.

This is the basic user-to-user delegation authorization rule. It means that a member of the role r (or a member of any role that is senior to r) can assign a user whose current membership satisfies prerequisite condition cr to role r (or a role that is junior to r) without exceeding the maximum delegation depth n.

Constraints Specification. In order to represent role-based privilege management constraints, we define rules that are extremely suited for constraints specification as well as enforcement. We articulate several constraints and specify them using a rule-based language introduced in [21].

A **static separation of duty (SSOD): incompatible roles assignment constraint** states that no common user can be assigned to conflicting roles in the incompatible role set $ira = \{r_1, r_2, \ldots\}$. This constraint can be represented as:

$cannot\_assign(u, r) \leftarrow$
$senior(r, one\_element(ira))\&$
$member\_of(u, one\_element(all\_other(ira, one\_element(ira))))$.
where $u \in U$, $r \in R$, *and* $ira \in IRA$.

The rule says if r equals one element of a set of the incompatible role assignments ira, and a user u is already member of another role other than r in the incompatible role set, then u cannot be assigned role r.

An **incompatible users constraint** states that two conflicting users in the incompatible user set iu=$\{u1, u2, ...\}$ cannot be assigned to the same role. This constraint can be represented as:
$cannot\_assign(u, r) \leftarrow$
$equals(u', one\_element(all\_other(iu, u)))\&$
$member\_of(u', r)$.

An **incompatible permissions constraint** states that two conflicting permissions in the incompatible user set ip=$\{p1, p2, ...\}$ cannot be assigned to the same role. This constraint can be represented as:
$cannot\_assignp(r, p) \leftarrow$
$equals(p', one\_element(all\_other(ip, p)))\&$
$in(p', permissions\_role(r))$.

A **role cardinality constraint** states that a role can have a maximum number N of user members. This constraint can be represented as:
$cannot\_assign(u, r) \leftarrow$
$greater\_than(cardi(r), maxcardi(r) - 1)$.

A **user cardinality constraint** states that a user can be member of a maximum number N of roles. This constraint can be represented as:
$cannot\_assign(u, r) \leftarrow$
$greater\_than(cardi(u), maxcardi(u) - 1)$.
We have demonstrated how different constraints can be specified using rules.

5 Implementation Details

Our implementation leverages role-based delegation features and X.509 attribute certificate. We attempt to implement the proof-of-concept prototype implementation of our architecture. An overview of the preliminary architecture is shown in Figure 2.

It consists of a number of services and management agents together with the objects to be managed. The enforcement agents are based on a combination of roles and rules for specifying and interpreting policies. Since delegation and revocation services are only part of a security infrastructure, we choose a modular approach to our architecture that allows the delegation and revocation services to work with current and future authentication and access control services. The

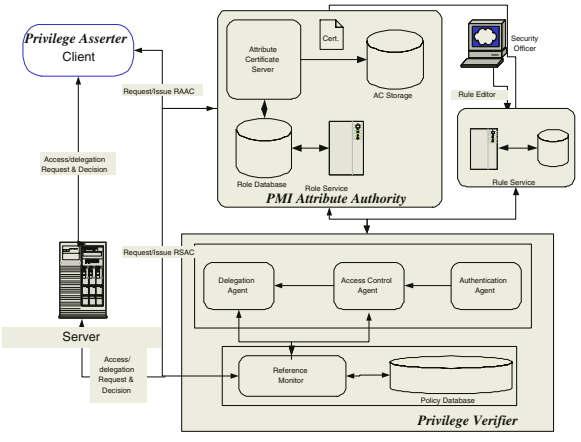


Fig. 2. Operational architecture for role-based EAM.

modularity enables future enhancements of our approach. The role service is provided by a role server and a role server maintains RBAC database and provides user credentials, role memberships, associated permissions, and delegation relations of the system. The rule service is provided by a rule server, which manages delegation and revocation rules. These rules are always associated with a role, which specifies the role that can be delegated. They are implemented as authorization policies that authorize requests from users. The rule editor is developed to simplify the management of these rules. As a portion of an integrated RBAC administration platform to manage various components, the rule editor is used to view, create, edit, and delete delegation and revocation rules. The delegation agent is an administrative infrastructure, which authorizes delegation and revocation requests from users by applying derivation authorization rules and processes delegation and revocation transactions on behalf of users. We implement these components as the delegation/revocation service: users' delegation/revocation requests are interpreted, authorized, and processed by the service; it creates RDM2000 elements based upon users' requests and maintains the integrity of the database by checking and enforcing consistency rules. The core of this service is a rule engine. We implemented the rule inference engine by extending SWI-Prolog [19] using its C++ interface. The rule engine has three functional units: a pre-processor, an inference engine, and a post-processor.

In Figure 2, three components are identified for managing attribute certificates: privilege asserter, privilege verifier, and PMI attribute authority as we described in Section 4. A privilege asserter is developed by using ActiveX control, named attribute certificate manager. The manager enables a user to import downloaded BER-encoded RAACs into Windows registry. Internet Information Server (Version 5.0) is used as a privilege verifier. An HTTP raw data filter, called AC filter, was developed using Microsoft ISAPI (Internet Server API) technology. An attribute certificate server was developed to generate *RAAC*s

and *RSAC*s. The programming library, called AC SDK, was built for supporting the functionality related to the generation of the attribute certificates. Netscape Directory Service 5.0 was used for both a role database and an AC storage. We also developed an application working as an access control policy server. This application has been developed in C++. An engine for making access control decisions is a major component in this application.

6 Conclusion and Future Works

Authentication mechanisms have been practiced at considerable length and various authentication schemes have been widely accepted. Unlike authentication mechanisms, privilege management which can conveniently enforce various business rules from different authorization domains among various applications still need to be investigated. In this paper, we have discussed issues of privilege management. We also attempted to utilize an existing delegation framework and attribute certificates in PMI. In addition, we demonstrated the feasibility of our architecture through a proof-of-concept implementation. We believe that this work would lead Internet-based applications to consider privilege management as a core component in their design and deployment.

Acknowledgements

This work was partially supported at the Laboratory of Information of Integration, Security and Privacy at the University of North Carolina at Charlotte by the grants from National Science Foundation (NSF-IIS-0242393) and Department of Energy Early Career Principal Investigator Award (DE-FG02-03ER25565).

References

1. G. Ahn, R. Sandhu, M. Kang, and J. Park. "Injecting RBAC to secure a Web-based workflow system," In *Proceedings of 5th ACM Workshop on Role-Based Access Control*. Berlin, Germany, July 2000.
2. E. Barka and R. Sandhu. Framework for role-based delegation model. In *Proceedings of 23rd National Information Systems Security Conference*, pages 101–114, Baltimore, MD, October 16-19 2000.
3. E. Bertino, E. Ferrari and V. Atluri. The specification and enforcement of authorization constraints in workflow management systems. *ACM Transactions on Information and System Security*, Vol.2 No.1, p.65-104, Feb. 1999
4. ITU-T Recommendation X.509. Information Technology: Open Systems Interconnection - The Directory: Public-Key And Attribute Certificate Frameworks, 2000. ISO/IEC 9594-8:2001.
5. S. Farrell and R. Housley. An Internet Attribute Certificate Profile for Authorization, PKIX Working Group, June 2001.

6. D. Ferraiolo, J. Cugini, and D.R Kuhn. "Role Based Access Control: Features and Motivations," In *Annual Computer Security Applications Conference*, IEEE Computer Society Press, 1995.
7. M. Gasser and E. McDermott. An Architecture for Practical Delegation a Distributed System. In *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy*, Oakland, CA, May 7-9,1990.
8. A. Hagstrom, S. Jajodia, F. P. Presicce, and D. Wijesekera. Revocations - a classification. In *Proc. 14th IEEE Computer Security Foundations Workshop*, pages 44–58, Nova Scotia, Canada, June 2001.
9. S. Jajodia, P. Samarati, V. Subrahmanian, and E. Bertino. A unified framework for enforcing multiple access control policies. In *Proceedings of the ACM SIGMOD international conference on management of data*, pages 474–485, 1997.
10. N. Li and B. N. Grosof. A practically implementation and tractable delegation logic. In *Proceedings of IEEE Symposium on Security and Privacy*, May 2000.
11. OSF DCE 1.0 Introduction to DCE, Open Software Foundation, Cambridge, MA, 1999.
12. J. Park, R. Sandhu, and G. Ahn. "Role-based Access Control on the Web," *ACM Transactions on Information and System Security*, 4(1), February 2001.
13. John Pescatore. Extranet Access Management Magic Quadrant, *Gartner Research Note (ID: M-13-6853)*, Gartner INC., May 2001.
14. R. S. Sandhu. Lattice-based access control models. *IEEE Computer*, 26(11):9–19, November 1993.
15. Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, February 1996.
16. R. Sandhu. Rationale for the RBAC96 family of access control models. In *Proceedings of the 1st ACM Workshop on Role-Based Access Control*. ACM, 1997.
17. D. Shin, Gail-J. Ahn, and S. Cho. Role-based EAM Using X.509 Attribute Certificate. In *Proceedings of Sixteenth Annual IFIP WG 11.3 Working Conference on Data and Application Security*, King's College, University of Cambridge, UK July 29-31, 2002.
18. M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson, and A. Essiari. "Certificate-based Access Control for Widely Distributed Resources," In *Proceedings of the 8th USENIX Security Symposium*, Washington, D.C., August 1999.
19. Wielemaker. "J. SWI-Prolog," http://www.swi.psy.uva.nl/projects/SWI-Prolog/
20. N. Yialelis, E. Lupu, and M. Sloman. Role-based security for distributed object systems. In *Proceedings of the IEEE Fifth Workshops on Enabling Technology: Infrastructure for collaborative enterprise*. IEEE, 1996.
21. L. Zhang, Gail-J. Ahn and B. Chu. A Rule-Based Framework for Role-Based Delegation and Revocation. *ACM Transactions on Information and System Security*, Vol.6, No.3, August 2003.