Authentication and Authorization User Management within a Collaborative Community

SYLVIA ENCHEVA
Stord/Haugesund University College
Department Haugesund
Bjørnsonsg. 45, 5528 Haugesund
NORWAY

SHARIL TUMIN
University of Bergen
IT-Department, 5020 Bergen
P. O. Box 7800
NORWAY

Abstract: Enterprise Identity Management (IDM) systems promise users to be equipped with a single enterprise identity and administrators with better users management environment. That increases security, provides the organizations with value-added data and money saving in user management. In fact, in order to do collaboration among independent organizations, each participant must first have a working IDM installment. This work discusses improvement of user management in networked systems. Group-role relations are used to facilitate users with an access control and permissions to different resources.

Key–Words: Enterprise identity management, security

1 Introduction

Computer-based access control can prescribe not only who or what process may have access to a specific system resource, but also the type of access that is permitted, [9]. In Role-Based Access Control (RBAC), access decisions are based on an individual's roles and responsibilities within the organization or user base, ([1], [3], and [10]).

Educational organizations, using a system that requires users' authentication and authorization data to reside locally in their system database, have to export their users' data to that system. This will involve a complicated data synchronization mechanism and creates operational issues that might compromise security for the service provider organization and for the client organization.

Our model simplifies user management in a large networked system by creating distributed groups for each role. The framework provides distributed user-group management and role-resource management. Organizations that are members of such a system share their users and groups data across all the organizations through a common communication framework.

What this mean in practice is that, each client organization manages its own users and groups, i.e. users-identification, password pairs and users group memberships, for authentication and authorization purposes within the collaborative community. Each provider organization manages its own authorization, i.e. role, resource, permission triplet and remote groups' roles memberships. A permission can be either positive (allow) or negative (disallow). A role with a negative permission signifies a container for blacklisted users. An authenticated user is authorized to access a resource if he/she is a member of a group and if the group is a member of a role and the role has positive permissions on that particular resource.

One needs a certain capabilities to perform one's duty. In some special cases or situations, these capabilities are in conflict with the laid down security policies. For example, two roles have been specified as mutually exclusive and cannot both be included in a user's set of authorized roles. Separation of duty requires that for any particular set of transactions, no single individual would be allowed to execute all transactions within that set.

Since users' management is done on independent sites, it is difficult to guarantee the uniqueness of users across inter-organizational boundaries. A person can be affiliated with many organizations at the same time. This problem is difficult to solve and may not be a major issue if a conflict of interests can be resolved in a role-group relationship.

The rest of the paper is organized as follows. Related work may be found in Section 2. The main results of the paper are placed in Section 3 and Section 4. The system architecture is described in Section 4.2 and Section 4.3. The conclusion is placed in Section 5.

2 Related Work

A formal model of RBAC is presented in [8]. Permissions in RBAC are associated with roles, and users are made members of appropriate roles, thereby acquiring the roles permissions. The RBAC model defines three kinds of separation of duties - static, dynamic, and operational. Separation of duties was discussed in [4], [9], and [15]. A framework for modeling the delegation of roles from one user to another is proposed in [2]. A multiple-leveled RBAC model is presented in [6]. The design and implementation of an integrated approach to engineering and enforcing context constraints in RBAC environments is described in [11], [16]). A location and time-based RBAC model is constructed by [7]. Although quite interesting the paper does not address industry related problems.

While RBAC provides a formal implementation model, Shibboleth ([14]) defines standards for implementation, based on OASIS Security Assertion Markup Language (SAML). Shibboleth defines a standard set of instructions between an identity provider (Origin site) and a service provider (Target site) to facilitate browser single sign-on and attribute exchange. Our work differs from Shibboleth in modeling, implementation and management of user/group/role. Shibboleth invests heavily on Java technology and SAML standards.

Our model is more open-ended based on Representational State Transfer (REST), a simpler Web-services implementation written in Python. The Origin site manages user and group memberships of users while the Target site manages permissions and role memberships of groups. The Origin site provides Web-services callable from Target sites to facilitate authorization on a protected resource. Additional needed procedures come to being by mutual agreement between sites.

RSA algorithm was publicly introduced in [12]. RSA is an asymmetric keys cryptography, also known as public key cryptography. We use RSA to digitally sign the calculated hash values by applying Message-Digest algorithm 5 designed in [13], and a hash function applied to the data block on the XML-WEB reply messages.

3 Collaboration among Independent Organizations

The foundation on which collaboration is built is trust. In order to collaborate, independent organizations must trust users from other organizations and authorization data associated with these users. Participating organizations must trust each other users and groups management processes. This is by no mean an easy thing to do, where political works are more important then technological.

Even today, many people within a significant number of large organizations are plagued with the burden for the need of to remember multiple user-identifiers and passwords. Many inactive users accounts remain enable long after theirs' period. Many user accounts are not related to actual persons within an organization. Many accounts are shared among many persons, which make accountability to be a problem. Such environments are security nightmares for any responsible security administrator.

We propose a model that simplifies user management in a large networked system by creating distributed groups for each role. One can add or remove users from roles by managing their membership in corresponding groups. The security model is based on:

1. Users are associated with groups.
2. Groups are associated with roles.
3. Roles are associated with resources' permissions.
4. Group-Role relationships provide users with an access control and permissions for a resource.

3.1 Managing Identities

Enterprise Identity Management (IDM) systems promise users to be equipped with a single enterprise identity and administrators with better users management environment. That increases security, provides the organizations with value-added data and money saving in user management. In fact, in order to do collaboration among independent organizations, each participant must first have a working IDM installment.

In order to collaborate, independent organizations must trust each other users' masses. A provider organization must trust a user authenticated by a client organization, and must grant access to its resources to that user authorized by that particular organization. A working IDM will help to increase the quality of users database of a particular organization, which in turn will enhance trust relationship with other participant organizations in the cooperation.

The IDM in an organization provides a centralized account identity database from which users who belong to that organization authenticate themselves. An account must necessarily link back to a real person associated with that organization. With this view in mind, it is desirable that a person owns one user account only.

The IDM must provide a centralized database for the users' groups membership data. Most of these data are created automatically from authoritative data source, for example from Human Resource (HR) system and Student Information System (SIS). The IDM must contain rules regarding accounts creation and destruction, accounts provisioning and users membership policies.

The IDM must also support delegation of management processes from a central authorities to local managers located at different units. The technological platform that supports the IDM system must be flexible enough to accommodate the rapid changes introduce by the real world in which it operates.

3.2 Ranking of Roles

Duties imply capabilities. This translates to one right of access and a right of actions upon a protected resource. This collaborative management model can be used by a security administrator to enforce a policy of separation of duties. Separation of duties appears to be of great value in a case of collaboration among various job-related capabilities where, for example, two roles have been specified as mutually exclusive and cannot both be included in a user's set of authorized roles. Separation of duty also requires that for any particular set of transactions, no single user would be allowed to execute all the transactions within the set. A system administrator can control access at a level of abstraction that is natural to the way those enterprises typically conduct business. This is achieved by statically and dynamically regulating users' actions through the established rules set, and the definition of roles, relationships, and constraints.

A static separation of duty enforces all mutually exclusive roles at the time an administrator sets up role authorizations, while a dynamic separation of duty enforces all rules at the time a user selects roles for a session. The dynamic separation of duty places constraints on a simultaneous activation of roles. A user can become active in a new role only if the proposed role is not mutually exclusive with any of the roles in which the user is currently active.

Defining disjoint groups' permissions is a duty of role managers at service provider domains and assigning users to proper groups is a duty of group managers at service client domains. These managers need to cooperate very closely. Policies and rules governing a resource's usage must be documented and understood by all parties. The managers at service provider domains have the right and the means to block any user in an event of a conflict.

A dynamic separation of duty requires that a user cannot hold two conflicting roles in the same session, e.g., an examine and an examiner of the same subject. We propose the use of a rank number in the range of 0 to 100, where role with a rank number 0 is the most capable and role with a rank number 100 is the least capable. Thus, a role with more permissions will have a lower rank number than a role with less permissions. A conflict of interests constrain must be checked by the application. An audit track of user assigned roles can be used to expose conflicts. In the event of conflicting roles, a user will get the least (i.e. higher rank number) role assignment automatically by the system.

4 System Supporting Collaboration

To support collaboration among independent organizations and distributive management model, the propose system must provide a framework in which all participating organizations can easily exchange users data and sharing resources. Organizations participate in the collaboration must effectively share their authentication and authorization data in real time.

The system must provide a framework in which participating servers can exchange messages and status information through service points in the form of Web applications and Web services for dynamic checking of:

- User credentials. Web based sign-on application.
- User authentications. Web based single-sign-on and session mechanism.
- User group and role relationship. Real time database queries for membership.
- User permissions. Real time database queries and conflict rule set.

In addition to above mentioned applications and services, the system must support Web based applications for distributed management and database for: User accounts, Users groups, Roles, Resources, and Permissions.

Collaboration among organizations entails that all of them must agree on the names of the groups to be used in user-group and group-role relations. The client organizations managers inform the provider organization managers about the group names when they subscribe for a particular resource. The provider organization managers bind then the group names to a particular role. A group name acts as a bridge in an inter-organization authorization mechanism. All users and groups are identified using the domain-name of their organization.

4.1 System Model

The client organization (CO) manages its domain users and groups membership, preferably using enterprise IDM. The provider organization (PO) manages roles, resources and permissions. At the provider organization, the roles membership of the remote groups from client organizations defines what type of permission a remote user can have on a particular protected resource.

For each shared resource the resource owner, at the provider organization needs to communicate the name of all the published resources to all potential subscriber organizations. This can be accomplished by providing descriptions of all resources in an Extensible Markup Language (XML) document at a Uniform Resource Identifier (URI) collectively agreed before hand by all partners.All subscribers can consult this XML file at any time for information pertinence to all published resources from a particular provider organization.

A provider organization defines groups for the roles relationships. These remote groups are associated with the Domain Name System (DNS) and domain names of clients' organizations. Examples of DNS domain names are 'uib.no' and 'hsh.no'. The resource provider also needs to maintain a subscribers' service points table. The service points table contains service description and Uniform Resource Identifier (URI) of client organizations Web services, from which the provider organization can send queries, for example, concerning user session and user group membership.

Within this model, both the provider organizations and the client organizations need to provide Web services for each other in order to communicate messages for data and control. There are many ways of providing these services, where among most common ones are, Java based remote method invocation (RMI), XML remote procedure (XML-RPC) and Simple Object Access Protocol (SOAP). We propose a simpler mechanism inspired by Representational State Transfer (REST).

Compare to the other mechanism, REST is much simpler because REST doest not defined its own transport protocol. REST depends on generic Web interface of HTTP GET, POST, PUT and DELETE. In order to provide and utilize Web services one need only to deploy Web server and client that support XML documents formatting and parsing.

4.2 System Architecture

To support REST type Web services participating organizations need to install and maintain a so-called three-tier architecture - Presentation Tier, Application Tier, andData Tier.

The presentation tier provides interfaces to the system. We propose using Apache Web server for implementing the presentation tier. These interfaces are referred to applications' clients by using some published URIs. A client can query a service of an application by addressing its URI and using HTTP GET parameters.

The application tier implements system processing logic. Data submitted by client queries trigger processing events in the application tier. Depending on URI, programs or scripts in the application tier will be executed. The GET parameters will be used as input parameters to these programs that will provide a response or an error message. We propose use of 'mod_python' for the Apache Web server. The application server will be programmed using Python scripting language. The module 'mod_python' extends the Apache capabilities by incorporating Python runtime interpreter within the Web server itself. All the Apache's application programming interface (API) are directly connected to the Python runtime environment, which make Web application programming simpler.

The data tier implements data store for the system. Information is stored and retrieved from a relational database management system. We propose using Oracle XE, a free small relational database management system form Oracle. It will then easy to upgrade the database to enterprise level, if that need arises. The relational database management system chosen should support Structured Query Language (SQL) standard. The database store long-term persistence information about users, groups, resources and applications states. Data stored in the database will affect the behavior of the whole system. Applications initialize and modify information in the database. Data is written, replaced and deleted by application programs. There exist many Python Oracle modules to choose from that integrate the application tier to the data tier in the system. We propose using DCOracle2 module.

4.3 System Implementation

We propose a communication mechanism based on REST, for message exchange between servers. An authentication server at a CO must support a single-sign-on (SSO) service for its domain users. The SSO service will support Web based sign-on application using Web browser cookies and database to store users' sessions. An active authenticated user on a particular workstation needs not to do multiple sign-on when accessing different resources at different POs.

An application portal at a PO must support user-authorization services. The application portal will check user's authorizations before it permits a user to access a resource or application provided by the provider organization. The application portal uses Web browser cookies techniques and database to store users' current authorization data. By using cookies and redirect, an authenticated user can be transferred from application portal to shared Web resource.

The central issue in implementing the system is on how the XML-WEB response messages from the Web services are formatted and how security is implemented. The response message must contain pertinence information in relation to a particular request and information that can be used to ensure message authenticity and integrity.

The security of the signed hash messaging system is base on the facts that: hash function - MD5 (Message-Digest algorithm 5) checksum of a data block provides some assurance that a transferred message has arrived intact. digital signatures - the checksum signed with a responder's private key can be verified by anyone who has access to the responder's public key. This proves that the sender signed it and that the message has not been tampered with.

This is used to ensure authenticity of the reply message really originated from the responder and not being modified in transit. The data block is not encrypted, so this technique does not protect privacy. However, the whole request/reply process can be done over secure http (HTTPS).

We used 'tlslite' Python module to implement RSA asymmetric keys cryptography in the system. By using 'tlslite.utils' function 'keyfactory' a private/public keys pair can be easily produced.

We propose providing XML-WEB response containing security information together with the reply. Our proposal is based on the following request/response scenario:

The requester uses basic authentication to send query of a particular service. The responder parses GET parameters and executes appropriate program if parameters are valid and the requester belongs to a list of valid clients, by checking its IP-address. The responder formats the reply block in XML containing a message identifier, a timestamp and data blocks. The responder then calculates a hash value for this reply block. The hash is then signed using its private key. A complete XML is sent back to the requester. The requester collects the reply block and verifies that the hash is valid by using the responder's public key. The requester parses the XML and execute appropriate program.

5 Conclusion

This collaborative management model can be used by security administrators to enforce a policy of separation of duties. Since users' management is done on independent sites, it is difficult to guarantee the uniqueness of users across inter-organizational boundaries. A person may be affiliated with many organizations at the same time. This problem is difficult to solve and may not be a major issue if conflicts of interests can be resolved in a role-group relationship. A split management of users, groups, roles, and permissions is proposed as a possible solution. By using simple XML-WEB communication mechanism, system based on REST Web services can be implemented. The coordinating works will be on how to define the common format for XML-WEB responses for each supported Web service. The optional usage of asymmetric keys sign/verify data block hash add security feature to XML-WEB responses, but will not enforce the use of it on clients that can not or will not make use of these security features. The propose system is open-ended and is not locked to any special Web framework, database and software tools. By using simple mechanism, the proposed system can accommodate complex interplay between several servers spanning collaborative net of many organizations serving community of users with shared resources.

References:

[1] Andress, M.: Access control. *Information security magazine*, April, 2001

[2] Barka, E., Sandhu, R.: Role-based delegation model/ hierarchical roles. *20th Annual Computer Security Applications Conference*, Arizona 2004

[3] Barkley, Beznosov, and Uppal: Supporting relationships in access control using Role Based Access Control,*Fourth ACM Workshop on Role-Based Access Control* 1999

[4] Bertino E., Bonatti, P.A., Ferrari E.: TRBAC: A temporal Role-Based Access Control model. *ACM Tr. on ISS*, 3(3), 2001, pp. 191–223

[5] Bhatti, R., Bertino E., Ghafoor A., Joshi, J.B.D.: XML-based specification for Web services document security. *IEEE Computer* 37(4) 2004

[6] Chou, S-C.; L^nRBAC: A multiple-levelled Role-Based Access Control model for protecting privacy in object-oriented systems. *Journal of Object Technology* 3(3), 2004, pp. 91–120

[7] Chandran, S.M., and Joshi, J.B.D. LoT RBAC: A Location and Time-based RBAC Model. *Proceedings of the 6th International Conference*

on Web Information Systems Engineering. New York, 2005

[8] Ferraiolo, D., Cugini, J., Kuhn., D. R.: Role-Based Access Control (RBAC): Features and motivations. *1995 Computer Security Applications Conference* 1995, pp. 241–248

[9] Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn R.D., Chandramouli R.: Proposed NIST standard for Role-Based Access Control. *ACM Transactions on Information and System Security (TISSEC)* 4(3) 2001, pp. 224–274

[10] Ferraiolo, D., Kuhn., D. R., and Chandramouli R.: Role-Based Access Control. Artech House, *Computer Security Series*, 2003

[11] Mattas A., Mavridis I., Ilioudis C., and Pagkalos I. Dynamically Administering Role Based Access Control, *WSEAS Transactions on Information Science and Applications*, 3 (10), 2006, pp. 1777–1784.

[12] Rivest, R. L., Shamir, A., and Adelman, L. M.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications ACM*, 21(2), 1978, pp. 120–126

[13] Rivest, R. L.: On NIST's Proposed Digital Signature Standard. *Proceedings of ASIACRIPT 1991*, 1991, pp. 481–484.

[14] http://shibbolethinternet2.edu

[15] Simon R., M. Zurko M.: Separation of duty in role-based environments. *Proceedings of 10th IEEE Computer Security Foundations Workshop*, Rockport, Mass., June 1997, pp. 183–194.

[16] Strembeck, M., Neumann, G.: An integrated approach to engineer and enforce context constraints in RBAC environments. *ACM Transactions on Information and System Security*, 7(3) 2004, pp. 392–427.