RBAC on the Web by Smart Certificates*

Joon S. Park and Ravi Sandhu

Laboratory for Information Security Technology
Information and Software Engineering Department
George Mason University

Abstract

We have described in another paper how to develop and use *smart certificates* by extending X.509 with several sophisticated features for secure attribute services on the Web. In this paper, we describe an implementation of RBAC (Role-Based Access Control) with role hierarchies on the Web as one possible application of smart certificates. To support RBAC, we issued smart certificates - which hold the subjects' role information - and configured a Web server to use the role information in the certificate instead of identities for its access control mechanism. Since the subjects' role information is provided integrity, the Web server can trust the role information after authentication and certificate verification by SSL, and uses it for role-based access control. To maintain compatibility with existing technologies, such as SSL, we used a bundled (containing the subject's identity and role information) smart certificate in the user-pull model.

1 Introduction

The World-Wide-Web (WWW) is a critical enabling technology for electronic commerce and business on the Internet. Its underlying protocol, HTTP (HyperText Transfer Protocol), has been widely used to synthesize diverse technologies and components in Web environments. WWW is commonplace. Increased integration of Web, operating system, and database system technologies will lead to continued reliance on Web technology for enterprise computing. However, current approaches to access control on Web servers are mostly based on individual users; therefore, they do not scale to enterprise-wide systems.

A successful marriage of the Web and a strong and efficient access control technology has potential for considerable impact on and deployment of effective enterprise-wide security in large-scale systems. Role-based access control (RBAC) [San98] is a promising technology for managing and enforcing security in large-scale enterprise-wide systems. The basic notion of RBAC is that permissions are associated with roles, and users are assigned to appropriate roles. This greatly simplifies security management. We were motivated by the need to manage and enforce the strong and efficient access control technology of RBAC in large-scale Web environments.

Public-key infrastructure (PKI) has been recognized as a crucial enabling technology for security in large-scale networks. To support PKI, X.509 [HFPS98, ITU93, ITU97] certificates have been widely used. The basic purpose of X.509 certificates is simply the binding of users to keys. Therefore, we have developed *smart certificates* by X.509 with several sophisticated features for secure attribute services, and introduced their possible applications on the Web [PS99].

In this paper, we describe an implementation of RBAC (Role-Based Access Control) with role hierarchies on the Web as one possible application of smart certificates. In the implementation, we used a Netscape Certificate server to issue smart certificates, and a Microsoft IIS 4.0 in Windows NT platform to support RBAC on the Web. However, this approach is also possible using other certificate servers or Web servers in different platforms by proper configuration. To maintain compatibility with existing technologies, such as SSL, we used a bundled (subject's identity and role information) smart certificate in the user-pull model [PS99].

*This work is partially supported by the National Security Agency at the Laboratory for Information Security Technology at George Mason University.

All correspondence should be addressed to Ravi Sandhu, ISE Department, Mail Stop 4A4, George Mason University, Fairfax, VA 22030, `sandhu@isse.gmu.edu`, `www.list.gmu.edu`.

The rest of this paper is organized as follows. First, in Section 2, we describe the technologies most relevant to our work, such as RBAC, X.509, and SSL. In Section 3, we give a quick overview of smart certificates. In Section 4, we describe how we actually implemented RBAC on the Web using smart certificates. This is followed by a discussion of related work in Section 5 and our conclusions in Section 6.

2 Related Technologies

2.1 Role-Based Access Control

Role-based access control (RBAC) has rapidly emerged in the 1990s as a promising technology for managing and enforcing security in large-scale enterprise-wide systems. The basic notion of RBAC is that permissions are associated with roles, and users are assigned to appropriate roles. This greatly simplifies security management.

A role is a semantic construct forming the basis of access control policy. With RBAC, system administrators can create roles, grant permissions to those roles, and then assign users to the roles on the basis of their specific job responsibilities and policy. Therefore, role-permission relationships can be predefined, which makes it simple to assign users to the predefined roles. Without RBAC, it is difficult to determine what permissions have been authorized for what users.

RBAC is a promising alternative to traditional discretionary and mandatory access controls, and ensures that only authorized users are given access to certain data or resources. It also supports three well-known security policies: data abstraction, least-privilege assignment, and separation of duties.

2.2 Public-Key Certificate (X.509)

A public-key certificate is digitally signed by a certificate authority (a person or entity) to confirm that the identity or other information in the certificate belongs to the holder (subject) of the corresponding private key. If a message-sender wishes to use public-key technology for encrypting a message for a recipient, the sender needs a copy of the public key of the recipient. On the other hand, when a party wishes to verify a digital signature generated by another party, the verifying party needs a copy of the public key of the signing party. Both the encrypting message-sender and the digital signature-verifier use the public keys of other parties. Confidentiality, which keeps the value of a public key secret, is not important to the service. However, integrity is critical, as it assures public-key users that the public key used is the correct one for the other party. For instance, if an attacker is able to substitute his or her public key for the valid one, encrypted messages can be disclosed to the attacker and a digital signature can be forged by the attacker.

ITU (International Telecommunication Union) and ISO (International Organization for Standardization) published the X.509 standard in 1988 [ITU93], which has been adopted by IETF (International Engineering Task Force). X.509 is the most widely used data format for public-key certificates today and is based on the use of designated certificate authorities (CAs). An X.509 certificate has been used to bind a public-key to a particular individual or entity, and it is digitally signed by the issuer of the certificate (certificate authority) that has confirmed the binding between the public-key and the holder (subject) of the certificate.

2.3 Secure Socket Layer (SSL)

SSL (Secure Socket Layer [WS96]) was introduced with the Netscape Navigator browser in 1994, and rapidly became the predominant security protocol on the Web. Since the protocol operates at the transport layer, any program that uses TCP (Transmission Control Protocol) is ready to use SSL connections. The SSL protocol provides a secure means for establishing an encrypted communication between Web servers and browsers.[1] SSL also supports the authentication service between Web servers and browsers.

SSL uses X.509 certificates. Server certificates provide a way for users to authenticate the identity of a Web server. The Web browser uses the server's public key to negotiate a secure TCP connection with the Web server. Optionally, the Web server can authenticate users by verifying the contents of their client certificates.

3 Smart Certificates Overview

An attribute is a particular property of an entity, such as a role, access identity, group, or clearance. If the attributes of individual users are provided securely on the Web by security services (e.g., authentication, integrity, and confidentiality); we can use those attributes for many purposes, including attribute-based access control, authorization, authentication, and electronic transactions. A successful marriage of the Web and secure attribute services has potential for considerable

[1] In many cases, due to export restrictions from the United States, only weak keys (40 bits) are supported, but SSL technology is intrinsically capable of very strong protection against network threats.

impact on and deployment of effective enterprise-wide security in large-scale systems.

In response, we have developed *smart certificates* to support secure attribute services on the Web by extending X.509 with several sophisticated features, without losing compatibility with X.509. Details for motivation and techniques about smart certificates are described in [PS99]. The smart certificates are able to provide short-lived lifetime, attributes, multiple CAs, postdated and renewable services, and confidentiality services in PKI. According to the requirements of applications, some of these new features can be selectively used in conjunction with currently existing technologies.

Smart certificates support both user-pull and server-pull models [PS99]. A bundled (identity and attributes) smart certificate is useful for the user-pull model, since Web servers require both identity and attribute information from each user in the model. In contrast, the bundled certificate is not a good solution for the server-pull model, because identity and attribute come from different places in the model. In this case, an additional channel is required for attribute transfer between the attribute server and Web servers.

To use certificates, user cooperation is required. Whenever the user connects to a Web server - which requires a certificate from the client - the user needs to select a proper certificate among her available certificates, and present it to the server. Once Web servers install the same CA (Certificate Authority) certificate as an acceptable certificate under a certain policy, a certificate issued by the CA can be used in many Web servers (even in different domains). For instance, Alice's smart certificate - which has her credit card information - can be used in many Web sites in different domains for electronic commerce on the Web.

If we use a bundled smart certificate, a user's attribute and public-key information can be included in a single certificate. This provides simplicity for both the protocol itself and for certificate administration. When we need separate authorities for attributes and authentication services, each authority signs separately on the same basic certificate and corresponding extension field, which contains attribute information. This can happen multiple times on a basic certificate by different attribute authorities. Each attribute authority has independent control over the attributes he issued. Even though a smart certificate can support independent management for the public key information and attributes, the system management becomes simpler if there is only one authority controlling both sets of information.

4 RBAC Implementation by Smart Certificates

Current approaches to access control on Web servers are mostly based on user identities. A successful marriage of Web and RBAC (Role-Based Access Control [San98]) technology can support effective enterprise-wide security in large-scale systems.

Figure 1 shows a schematic of RBAC on the Web. The role server has user-role assignment information for the domain. After a successful user authentication, the user receives his or her assigned roles in the domain from the role server. Later, when the user requests access to a Web server with the assigned roles in the domain, the Web server allows the user to execute transactions in the server based on the user's roles instead of her identity. The Web servers may have role hierarchies or constraints based on their policies. Administration of the role server can be performed in a decentralized manner by administrators on the Web [SP98].

Nevertheless, the important question arises: how can the Web servers trust the role information presented by users? For instance, a malicious user may gain unauthorized access to the Web servers by using forged role information. Therefore, we must protect the role information from being forged by any possible attacks on the Web, as well as in the end-systems.

There are many possible ways to support the above requirement. In this paper, as one possible solution, we will describe how we implemented RBAC (Role-Based Access Control) with role hierarchy on the Web using smart certificates. In this implementation, we used a bundled (subject's identity and roles) smart certificate in the user-pull model [PS99], maintaining compatibility with existing technologies such as SSL, without requiring an additional channel for attribute transfer on the Web. We used a Netscape Certificate server and a Microsoft IIS 4.0 in Windows NT platform to support RBAC on the Web. However, this approach is also possible using other certificate servers or Web servers in different platforms.

4.1 Obtaining and Presenting Assigned Roles on the Web

Figure 2 shows how a bundled smart certificate is issued and used for RBAC on the Web. If a user, Alice, wants to execute transactions in the Web servers in an RBAC-compliant domain, she first connects to the role server in the beginning of the session. After the role server authenticates Alice, it finds her explicitly assigned roles in the URA (User-Role Assignment [SP98, SB97]) database and creates a smart

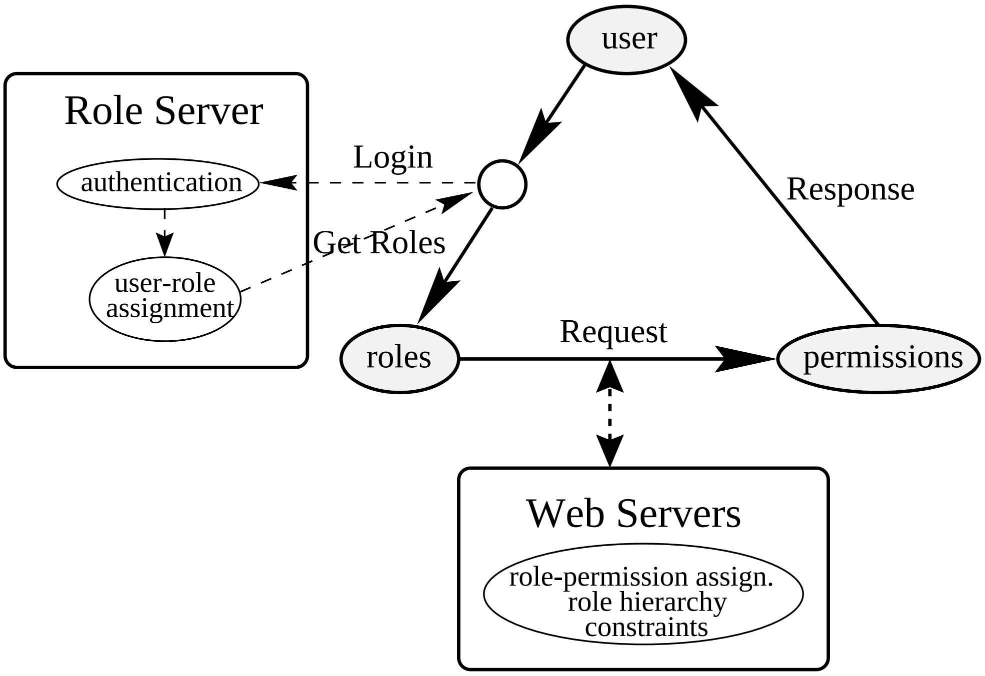

Figure 1: A Schematic of RBAC on the Web

certificate (which holds her explicitly assigned roles). Then, the smart certificate is sent to and stored in Alice's machine - which has Alice's private key corresponding to the smart certificate - so that Alice does not need to go back to the role server to obtain her assigned roles until the certificate expires. Consequently, she can use the roles in her smart certificate in the RBAC-compliant domain as long as the certificate is valid. In this implementation, we used the OU (Organization Unit) field in X.509 certificates to store each subject's role information, and both identity and roles are signed by a single certificate authority. However, if a smart certificate has different attributes (which need to be signed by different CAs), or obtains detailed attribute information, such as validity for each attribute or attribute issuer, we can use the extension fields of X.509. Furthermore, separate certificates for identity and roles are also possible in the server-pull model.

Alice may have many smart certificates in her machine. When Alice requests access to a Web server - which requires clients' certificates and has PRA (Permission-Role Assignment [SBC+97]) information - by typing the server's URL in her browser, the browser and Web server authenticate each other over SSL. After the browser receives and verifies the server's X.509 certificate, Alice needs to select a proper smart certificate - which has her role information - and sends it to the Web server. The Web server authenticates Alice by verifying the smart certificate. If the smart certificate is valid and verified successfully, the Web server trusts the role information in the certificate and uses it for RBAC with a role hierarchy and permission-role assignment information in the Web server, as described below.

4.2 RBAC in the Web Server

Internet Information Server (IIS) depends on Windows NT File System (NTFS) permissions for securing individual files and directories from unauthorized access. NTFS permissions can be precisely defined with regard to the users who can access the contents of the server and which permissions are allowed to the users, while Web server permissions are applied to all users accessing the Web server.[2] NTFS permissions apply only to a specific user or group of users with a valid Windows NT account.

In a Windows NT environment, we can control user access to the contents in a Web server by properly configuring the Windows NT file system and the security features of the Web server. When the user attempts to

[2]For instance, Web server permissions can control whether users visiting the Web site are allowed to view a particular page, run scripts, or upload information to the site.

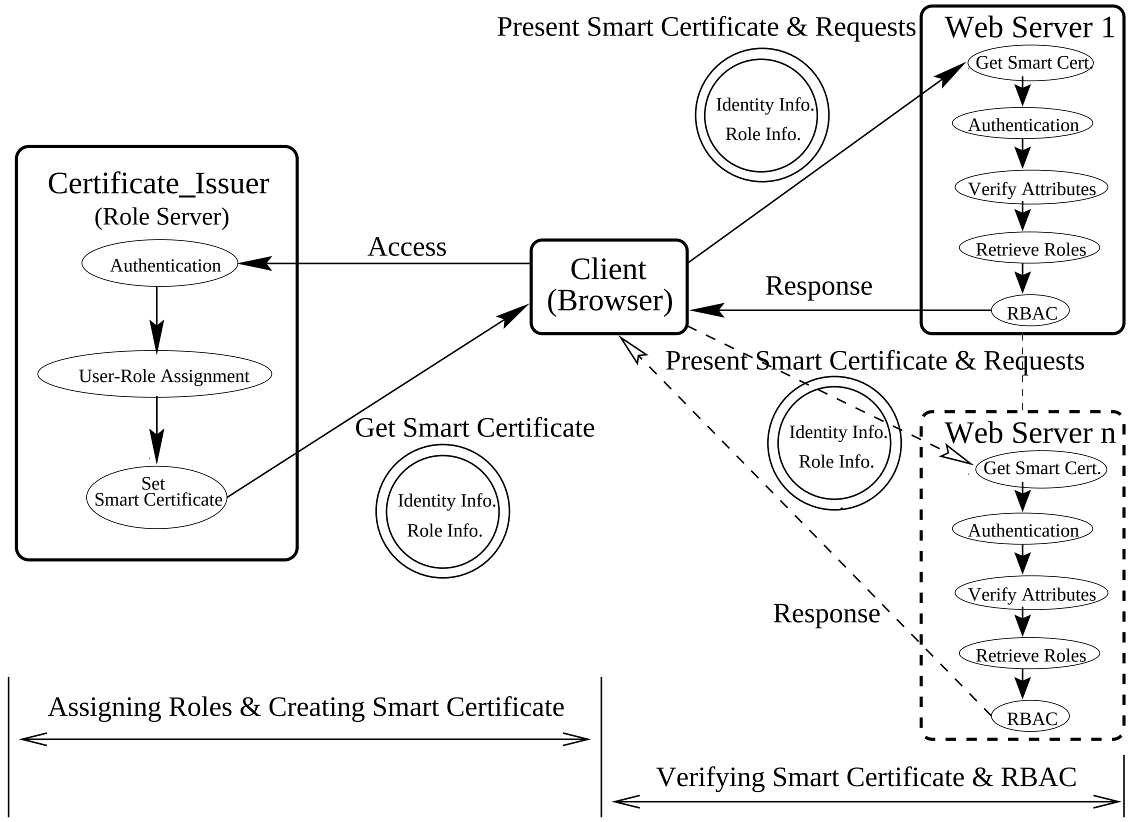

Figure 2: RBAC on the Web by Smart Certificate

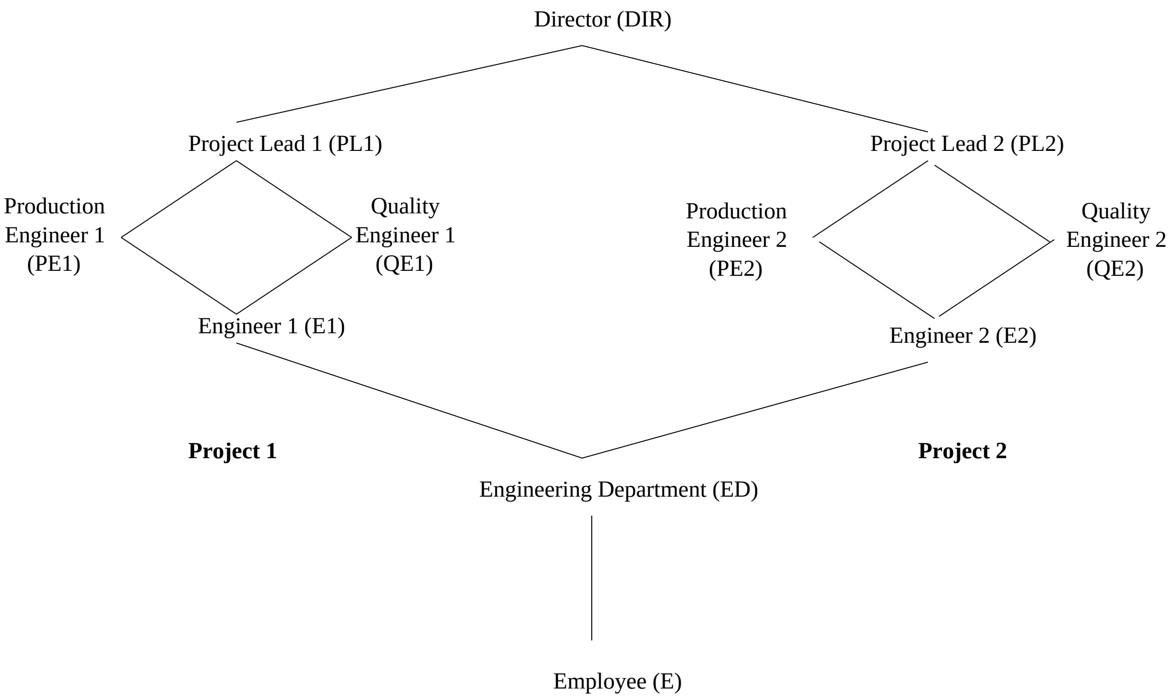

Figure 3: An Example Role Hierarchy

access the Web server, the server executes several access control processes to verify the user and determine the allowed level of access based on its policy.

To support RBAC with the role hierarchy depicted in Figure 3, we configured an IIS 4.0 with two creative ideas: *role accounts* and *PAA (Permission-Account Assignment)* in the Web server. These ideas are described in the following subsections.

4.2.1 Mapping Roles to Role Accounts

Since the Web server uses roles - denoted in the client smart certificates - for its access control mechanism, regular user accounts are not necessary in the server.[3] Instead, we created the role accounts (e.g., Director, Project_Lead1, Project_Lead2, Project_Engineer1, Quality_Engineer1, and so on) in the Windows NT server, where the Web server (IIS 4.0) is installed. Then, by configuring the Web server's certificate mapping feature, we mapped each role in the role hierarchy in Figure 3 to the corresponding role account in the Windows NT server. For example, we mapped the role DIR to the role account Director in the server. After a user (subject), Alice, authenticates to a Web server over SSL by sending her client smart certificate - which has the role "DIR" - to the server, she is mapped to the role account "Director" in the Windows NT server. As a result, even though Alice does not have an account in the server, she acquires the Director's permission in the server, since she is assigned to the role "Director," denoted in her smart certificate. The permission of each role account depends on the policy of the Web server.

[3] The Web server may need administrator accounts for its maintenance.

4.2.2 Providing Role Hierarchy

How then can the Web server support the role hierarchy? Figure 4 shows how we used a built-in access control mechanism in the Windows NT server to support the role hierarchy depicted in Figure 3. Reflecting the roles in the hierarchy, we created the role accounts, such as Director, Project_Lead1, Project_Lead2, Project_Engineer1, Quality_Engineer1, and others. We also created directories in the Windows NT file system, where each directory has files to be accessed by a specific role in the role hierarchy. Subsequently, we configured the Windows NT file system to assign each role account to specific access rights to the directories based on the role hierarchy. For instance, the role account Project_Lead1 is assigned to access rights to the Project_Lead1's directory - which has resources for the role Project_Lead1 - and the directories that

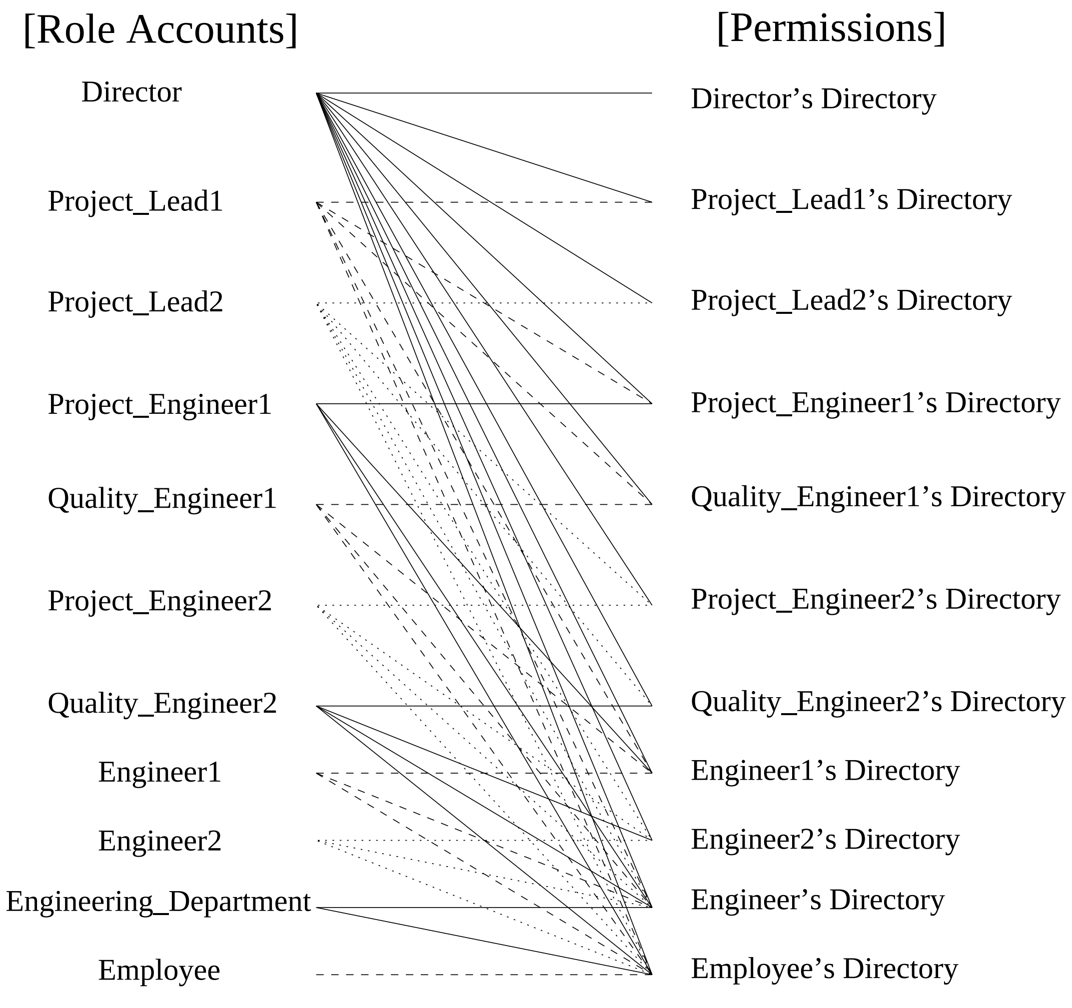

Figure 4: Permission-Account Assignment (PAA)

require the roles junior to the Project_Lead1 role in the role hierarchy. In other words, if Alice is mapped to the role account Project_Lead1, she obtains permissions assigned to the role account Project_Lead1, thereby acquiring access rights to the directories for Project_Lead1, Project_Engineer1, Quality_Engineer1, Engineer1, Engineering Department, and Employee.

As a result, after verifying the smart certificate, the Web server allows the user, Alice, to execute transactions based on her roles - contained in the OU field of the certificate - instead of her identity. In other words, the Web server does not care about the user's identity. This resolves the scalability problem of the identity-based access control, which is being used primarily in existing Web servers. Furthermore, since the Web server also uses a role hierarchy, it supports a natural means for structuring roles to reflect an organization's lines of authority and responsibility. Each Web server may have a role hierarchy different from that in other servers. The location of RBAC-compliant Web servers is geographically free from that of the role server, since smart certificates (which include the subjects' role information) can be issued by one certificate server for use by other Web servers, regardless of their physical location.

5 Related Work

5.1 *Secure Cookies*

Park and Sandhu[4] have developed *secure cookies* by extending the existing cookie mechanism between Web servers and browsers with cryptographic technologies, and also implemented RBAC with role hierarchies on the Web using secure cookies [PSG99]. To protect the role information in the cookies, they provided security services, such as authentication, confidentiality, and integrity, to the cookies using PGP and CGI scripts in the Web servers. The cookie-issuing Web server creates a set of secure cookies including the user's role information, and other Web servers use the role information for RBAC with role hierarchies after cookie verification. The use of secure cookies is a transparent process to users and applicable to existing Web servers and browsers.

5.2 *getAccess*

enCommerce has released *getAccess* [enC98] to implement a hierarchical role-based model for the organization online. Each role defines a specific access privilege to one or more resources. The roles can be grouped into *macro roles*, and macro roles can also have other macro roles. There are four main software modules in this product: registry server, access server, administration application, and integration tools. The access server is located in a company's Intranet or Extranet, while the registry server is always located in the Intranet. A user always connects to the access server first via browsers. The access server then connects the registry server to obtain the user's identification and roles through a secure connection. Subsequently, the registry server authenticates the user and returns the user's encrypted role information through cookies. These cookies are temporarily stored in RAM on the user's machine while the browser is open. When the user connects to a Web server in the Intranet, the browser sends the cookies to the Web server. The Web server then decrypts and uses the encrypted role information in the cookies for role-based access control in the server.

The *getAccess* mechanism uses encrypted cookies; however, there is a huge difference between its approach and *secure cookies*. The encrypted cookies are not stored in the user's machine after the session. In other words, if a session is ended by closing the browser, the encrypted cookies disappear. This means that whenever a user, Alice, needs to connect to a Web server with her roles, she must connect to the registry server first through the access server. On the contrary, secure cookies - which obtain the user's role information - can be stored in the user's machine securely after the session, even when the power of the user's machine is off. This is possible because the secure cookies can be provided integrity and authentication services as well as encryption. Therefore, once Alice obtains her secure cookies, she can use her roles until the cookies expire, without having to connect to the cookie issuer.

5.3 *TrustedWeb*

Siemens Nixdorf released *TrustedWeb* [Nix98], which supports role-based access control for Web contents and applications, as well as security services, such as mutual authentication, integrity, and confidentiality for Intranets. The system, combining elements from both Sieman's SESAME [PP95] and Kerberos [Neu94], provides a single list of users on its central domain security server and assigns roles to the users. Therefore, access to the individual Web servers in the Intranet is controlled based on the role rather than the identity of the user. However, to use *TrustedWeb*, the client's browser needs specific software installed in the client's machine to communicate with the *TrustedWeb* servers in the Intranet while our techniques do not require any specific

[4] The Laboratory for Information Security Technology (LIST) at GMU, http://www.list.gmu.edu

software in the client side.

6 Conclusion

In this paper, we have described how we implemented RBAC with role hierarchies on the Web using *smart certificates*. The certificate authority issues a smart certificate, including a subject's identity and role information, and Web servers use the role information for RBAC with role hierarchies after identity and attribute verification. This access control mechanism solves the scalability problem of existing Web servers. The implementation is transparent to users and applicable to existing Web servers and browsers.

References

[enC98] enCommerce. *getAccess*, 1998. http://www.encommerce.com/products.

[HFPS98] R. Housley, W. Ford, W. Polk, and D. Solo. *Internet X.509 public key infrastructure certificate and CRL profile*, September 1998. draft-ietf-pkix-ipki-part1-11.txt.

[ITU93] ITU-T Recommendation X.509. *Information technology - Open systems Interconnection - The Directory: Authentication Framework*, 1993. ISO/IEC 9594-8:1993.

[ITU97] ITU-T Recommendation X.509. *Information technology - Open systems Interconnection - The Directory: Authentication Framework*, 1997.

[Neu94] B. Clifford Neuman. Using Kerberos for authentication on computer networks. *IEEE Communications*, 32(9), 1994.

[Nix98] Siemens Nixdorf. *TrustedWeb*, 1998. http://www.sse.ie/TrustedWeb.

[PP95] Tom Parker and Denis Pinkas. *SESAME V4 - OVERVIEW*. SESAME Technology, December 1995. Version 4.

[PS99] Joon S. Park and Ravi Sandhu. Smart certificates: Extending x.509 for secure attribute services on the Web. In *Proceedings of 22nd National Information Systems Security Conference*, Crystal City, VA, October 1999.

[PSG99] Joon S. Park, Ravi Sandhu, and SreeLatha Ghanta. RBAC on the Web by secure cookies. In *Proceedings of the IFIP WG11.3 Workshop on Database Security*. Chapman & Hall, July, 1999.

[San98] Ravi Sandhu. Role-based access control. *Advances in Computers*, 46, 1998.

[SB97] Ravi Sandhu and Venkata Bhamidipati. The URA97 model for role-based administration of user-role assignment. In T. Y. Lin and Xiaolei Qian, editors, *Database Security XI: Status and Prospects*. North-Holland, 1997.

[SBC+97] Ravi Sandhu, Venkata Bhamidipati, Edward Coyne, Srinivas Ganta, and Charles Youman. The ARBAC97 model for role-based administration of roles: Preliminary description and outline. In *Proceedings of 2nd ACM Workshop on Role-Based Access Control*, pages 41–50. ACM, Fairfax, VA, November 6-7 1997.

[SP98] Ravi Sandhu and Joon S. Park. Decentralized user-role assignment for Web-based intranets. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*, pages 1–12. ACM, Fairfax, VA, October 22-23 1998.

[WS96] D. Wagner and B. Schneier. Analysis of the SSL 3.0 protocol. In *Proceedings of the Second UNIX Workshop on Electronic Commerce*, November 1996.