CERIAS Tech Report 2004-32

TOWARDS IMPROVED FEDERATED IDENTITY AND PRIVILEGE MANAGEMENT IN OPEN SYSTEMS

by Rafae Bhatti, Elisa Bertino, Arif Ghafoor

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

Towards Improved Federated Identity and Privilege Management in Open Systems

Rafae Bhatti, Elisa Bertino, Arif Ghafoor

Federated identity and privilege management are the cornerstones of access management on the Web. The increasing trend of business integration across enterprises and Web-based collaboration has led to tremendous growth of the identity and privilege management research and products in the recent past. However, despite the existence of available mechanisms, there are drawbacks in almost all well-known schemes that make them inadequate for use in large scale open system. Additionally, the migration of these mechanisms to the Web environment is happening at dissimilar pace, resulting in a wide gap in integrating privilege management with existing federated identity mechanisms to provide a comprehensive access management solution. In this paper, we discuss these issues in detail, namely the shortcomings of federated identity mechanisms, and their integration with privilege management mechanisms. In response, we provide an integrated approach to Web-based access management that combines a decentralized federated identity mechanism with a privilege management framework. Our solution allows name-binding to be avoided; doing so is essential to scalability and privacy in open systems. The solution has been prototyped and preliminarily tested to determine its feasibility.

1. Introduction

The highly-networked enterprise environment is characterized by strategic partnerships to seize better business opportunities on the Internet. The desire to capitalize on such opportunities has driven the demand for mechanisms that allow web-based collaboration between enterprises. The access management to enterprise resources in such collaborative environments is absolutely critical for their security. The major industrial players in security also opine that "today's collaborative and interconnected e-business landscape requires a secure and effective way for enterprises to share trusted user identities"[1] and entitlements. However, if not done properly, imprecise access management could adversely affect the level of un-interrupted interoperability needed to seamlessly integrate enterprise units and business processes. The ability to federate identity across organizations while maintaining access rights and privileges is thus a major challenge [1]. The solution is federated identity and privilege management, which now stands as the key to seamless and secure enterprise integration and collaboration on the Web. The federated identity and privilege management mechanisms of today are, however, not without their shortcomings which need to be overcome in order to ensure that these mechanisms scale well. Among them is the use of (i) a centralized approach to providing federated identity, and (ii) identity or capability-based credentials. The centralized approach to federated identity has been subject to much scrutiny in recent past, with specific references to the most widely used such scheme, Microsoft Passport [2], as shall be shortly discussed. Similarly, the drawbacks of identity and capability-based credentials used in most existing systems have also been reported in the literature, and are discussed in next section. In addition to these shortcomings, there is another concern that needs to be alleviated. The development of Web-based federated identity solutions has advanced at a much rapid pace as compared to the Web-based privilege management mechanisms. The growth of the former may be attributed to advances in biometrics and cryptographic tools that have quickly become marketable, whereas the commercial tools for the latter are still primitive and advanced solutions are mostly in research phase. The research community has recognized the fact that the interplay between identity and access management should be more carefully evaluated, and present access control models need to be appropriately refined [1]. However as it stands now, there is a wide gap in integrating privilege management with existing federated identity mechanisms to provide a comprehensive access management solution. This disparity is quite alarming, and the increasing trend of migrating enterprise operations to the Internet demands a significant evolution of the

[1] Federated Identity white paper, RSA Secuirty Inc.

traditional access management mechanisms in order to secure the inherently dynamic Web-based resources. Simply put, both federated identity and privilege management are cornerstones of an access management framework; a weakness in any one component would render any such framework inadequate for dynamic collaborative business environments. In this paper, we discuss these challenges, namely the shortcomings of federated identity mechanisms, and their integration with privilege management mechanisms. In response, we present an integrated approach to federated identity and privilege management specifically designed for Web-based platforms.

At the very onset, we would outline the requirements that we believe an integrated federated identity and privilege management mechanism should satisfy. The following sections would then build the necessary motivation behind these requirements and discuss how our proposed framework satisfies them.

(i) Single sign on (SSO): SSO is a fundamental component of federated identity, and allows for privilege management across enterprises in a manner transparent to the end user. It essentially implies persistence of user identity and entitlement across enterprise domains, and allows users within and across enterprises to seamlessly transfer their authorizations across multiple points of policy enforcement. Although many SSO solutions abound, the widening gap between identity and privilege management leads to many challenges with regards to granting single-sign-on access to collections of resources that might have contradictory access-protection rules [1].

(ii) Effective access control: The privilege management component of the access management solution relies on the strength of the access control model. A comprehensive access management solution should support an effective access control model that allows flexible and fine-grained access control to dynamically evolving enterprise resources. This requirement is particularly challenging to meet in a Web-based environment.

(iii) Decentralized model: This implies that the system should not rely on a centralized or single point for accessing user authentication and authorization information. Instead, this control should be distributed. This requirement is motivated by the market demand for B2B scenarios, where it is desired to have a decentralized model for federating user identities and entitlements and thereby avoiding a scenario where "one enterprise essentially authenticates the world population"[2].

(iv) Authentication for strangers: In the widely distributed Internet environment, it is no longer a workable business model for a service provider to assume advance knowledge of the identities or capabilities of all users. The use of identity and capability-based credential in most existing systems is a major bottleneck to achieving this objective.

(v) Trust, Anonymity and Privacy: Privacy protection is becoming an increasingly significant issue, more so from social and legal perspective, and it is a challenge to provide sufficient level of anonymity and privacy without compromising on security. The paradox here is clear: while avoiding name-binding appears viable for preserving privacy, it complicates the accountability in trust establishment.

(vi) Standardized Approach: With numerous schemes in several stages of adoption, it is only prudent to take an incremental or "integrate"-able approach: design new solutions that complement existing accepted standards. Standardization is a long, over-whelming process, and a new standard effort every often would not contribute positively to the existing mix of adopted and un-adopted specifications. We have therefore carefully evaluated the existing technologies and attempted to address only the open issues; for other functionality, we provide hooks within our specification where existing standards can be tied into.

[2] B. Pfitzmann, M. Waidner, "-Federated Identity Management Protocols-", IBM Zurich Research Labs, To Appear.

The remainder of the paper is organized as follows. We begin with a brief overview of federated identity and privilege management, and then provide a comprehensive survey of the research leading to the current state-of-the-art in both these areas. The survey thereby highlights the issues related to centralized nature and name-binding in existing schemes, motivates the need for meeting the above-listed requirements, and emphasizes the design of an improved solution. The following two sections describe the specification and software architecture of our proposed framework. Our specification is XML-based, and is captured through a context-free grammar called X-Grammar introduced in [12], which follows the same notion of terminals and non-terminals as in BNF, but supports the tagging notation of XML that also allows expressing attributes within element tags. The paper concludes with discussion of our prototype implementation and future work.

2. Background, Motivation and Related Work

In their current form, federated identity and privilege management solutions are an integral part of the access management framework in a collaborative enterprise environment. Although they have begun to gain popularity only recently, the concept behind them derives its motivation from the classical authentication and authorization protocols, as we shall discuss in this section. We acknowledge the work presented in [16, 17] as providing us a lead in some parts of this survey.

The various approaches presented in the literature have not always clearly separated authentication step from authorization, and hence we shall discuss both schemes together in their order of evolution. The initial approaches to distributed authorization relied on the distributed authentication schemes used in conjunction with access control lists for local authorization. A seminal work in authentication protocols based on symmetric-key cryptography has been presented in [3] and implemented as Kerberos [4]. Kerberos allows mutual authentication and secure communication over the network by the use of symmetric key encryption and authentication credentials issued by a centralized Kerberos server. Kerberos authentication credentials are based on identity, and are suited for use in identity-based authorization mechanisms such as access control lists. Such schemes have scalability problems in distributed systems vis-à-vis management of user identities and access rights which motivated our approach for decentralization and avoiding name-binding. Additionally, there emerge key management issues in symmetric key cryptography in widely distributed environments. As opposed to identity-based, capability-based approaches to access control have later been introduced in the context of operating systems [5-7] wherein the authorization decision is taken based on the key holder's stated capability. Credentials extend the notion of the capabilities by using additional cryptographic information, such as issuer-specific and principal-specific signatures, to ensure proper replication and selective revocation of compromised credentials, respectively. Various schemes have emerged for distributed authorization using credentials [8-10]. In [8], the X.509 certificate scheme for authentication is introduced. It is based on the Public Key Infrastructure (PKI) and binds a public key to a global name. Its later version introduces the X.509 Privilege Management Infrastructure (PMI) [11] which uses X.509 together with the notion of an access control credential called Attribute Certificate which binds a name to a set of privileges. In contrast to name binding, the approach taken in SPKI/SDSI and KeyNote [9, 10] is key-centric, i.e. the access control credential is directly bound to a public key with authorizations. In this case, the public key effectively identifies the principal without using global names, and the access decision is taken based on the access rights contained in the credential. The PKI-based approach to distributed access control is traditionally known as Trust Management (TM). We shall henceforth refer to the credentials used in TM schemes as TM credentials. In the schemes [8-10], the TM credentials used have their drawbacks. X.509–based TM credential is identity-oriented, and its name binding tends to be long-lived, making it ill-suited to expressing distributed authorizations. The use of key-centric TM credentials removes the dependency on names, and introduces the concept of globally unique keys. It hence achieves the goal of decentralization through delegation. However, the binding of access control credential with the key blurs the distinction between authentication and authorization, thereby tightly coupling the two. While an integrated approach to authentication and authorization may be desirable in

some situations, it is not always the most flexible and practical option. Such an approach limits the expressiveness of the access control mechanism. This limitation arises due to two reasons. Firstly, not all system-specific capabilities may be known in advance in a distributed environment and hence a capability-based credential is not suitable to expressing authorizations. This is especially the case if SSO is to be supported, because the intention there is to prevent having multiple authorization mechanisms for access to multiple resources. Secondly, the use of an access control credential embedded within an authentication scheme is not sufficient to meet the effective access control requirement outlined earlier.

The next generation of distributed authorization models has attempted to alleviate this drawback by designing effective and more expressive access control schemes. Many recent models have employed the Role Based Access Control (RBAC) as a solution to privilege management in large scale enterprise systems. RBAC has already been shown to be effective for privilege management on the Web [18, 19, 21]. We now evaluate the merits of existing RBAC-based approaches with regards to our requirements. The X.509 based PMI and its reference implementations such as PERMIS [20], is not suitable due to its name-binding approach. A work that attempts to address this issue is presented in [21]. Although they do not focus on authentication, their idea of using a "smart certificate" for role-based authorizations is appealing and could possibly be used to provide SSO. Another prominent specification is the XML-Based Access Control Markup Language (XACML) [22]. XACML has recently been adopted as a standard specification. However, XACML in its present form does not support role-based access control, and hence lacks the desirable features like simplified administration and privilege management in large scale enterprises. It also has no explicit support for strong authentication. X-GTRBAC and OASIS [12, 13] are similarly expressive models using RBAC to define dynamic fine-grained access control in an enterprise environment. However, both these schemes also do not provide explicit support for strong authentication. Additionally, they use either identity or capability-based credentials and are not scalable to the case of role assignment for unknown users on the Internet. Two approaches for role assignment to unknown users based on TM credentials have been presented in [14, 15]. The Trust Establishment Project (TEP) [14] uses a Trust Policy Language (TPL) to map holders of public key certificates to roles based on attribute contents thereof. A Role based Trust management (RT) framework is introduced in [15]. It merges features from TM and RBAC and uses a more expressive policy language compared to TPL. The TM credentials used in [14, 15] are examples of property-based credentials, as opposed to identity or capability-based, because they allow user authentication and subsequent authorization (i.e. role assignment) based on certain properties thereof. Referring back to our requirement related to authentication for strangers, these are the type of credentials that we need to authenticate unknown users into known roles, since pre-defined identities and capabilities cannot be assumed. Although they come one step closer to meeting our requirements, both schemes, however, have their shortcomings. While TEP and RT provide a TM credential-based mechanism to assist in distributed authorizations, they do not support an elaborate access control scheme beyond the basic permission-to-role assignment mechanism in RBAC. Additionally, TEP in its present implementation uses X.509-based PKI, and hence suffers from the name-binding problems discussed above. Despite the shortcomings, the use of TM credentials in RBAC setting is appealing for our purposes because it would allow us to integrate distributed authentication support within a well-accepted authorization mechanism, and essentially combine the features of the approaches [12-15].

In order to provide a complete federated identity and privilege management solution, however, we also need to satisfy the requirement of SSO. The most prominent Web-based SSO system in use today is the Microsoft Passport [2]. Passport is based on a centralized server model, and is much like a Kerberos counterpart for the Web. However, on an Internet scale, the centralized approach is not without its due share of risks- amongst them are compromise of the central repository and subjugation to denial of service attacks. A centralized model, in fact, is antithetical to the distributed nature of the Internet [23]. Therefore, the potential compromise of system security through the use of Passport as a SSO mechanism is unacceptable, and calls for a better approach. We however emphasize that SSO is only as effective as

the underlying authentication and authorization protocols, and those need to be improved to provide a more quality experience to the end user. This is where the motivation of our work lies; we address the problem of providing improved identity and privilege management solution through an interoperable and modular design of underlying authentication and authorization mechanisms. In particular, we integrate strong authentication and decentralized SSO support within an authorization model, while also cryptographically enhancing the latter with the support for issuing persistent authorization assertions to make the SSO more efficient. In the following sections, we provide the design and grammar specifications of our access management framework.

3. Proposed Solution

The emphasis of our proposed solution is to design and implement modular components to interface with an existing authorization model so as to extend it with the capabilities for federated identity and privilege management in open enterprise environments.

An initial requirement the authorization model need satisfy is suitability to Web-based applications. Based on the original system requirements and the discussion in Section 2, we believe that X-GTRBAC [12] is one candidate, and has therefore been adopted as the authorization model in our system. For the benefit of the reader, we tabulate the salient features of the model in Table 1. The X-Grammar specification is presented in Appendix A, whereas a detailed discussion of its access control mechanism is found in [12]. The central idea is that the system uses credentials supplied by users to assign them to roles (authentication) subject to any assignment constraints. The users can then access resources according to their role memberships (authorization) subject to any dynamic access constraints. Hence, X-GTRBAC supports fine-grained attribute-based access control with modular authentication and authorization mechanism. However, the model in its present form lacks strong authentication and persistence management. To provide this support, we outline the configuration shown in Figure 1. The persistence management and authentication modules can be distinct components with well-defined interfaces, and could possibly be published as Web services. This not only results in a scalable system, but also provides the flexibility of managing the core functionality of these components independently of each other. We emphasize that the modular architecture of the distributed authentication and authorization system allows interoperable access management across heterogeneous domains, and could realize the possibility of a decentralized SSO paradigm. This claim shall be supported with technical discussion in this section.

The X-GTRBAC model through its XML-based specification enables effective Web-based access control capabilities, which have been shown to be applicable in Web services [24] and enterprise systems [12]. That together with its decentralized administration model [25] makes it a promising candidate for access management in open systems. In addition, the initial framework presented in [19] leading to the X-GTRBAC model has been cited by the Organization for Advancement of Structured Information Standards (OASIS) in its announcement of the ratification of the ANSI RBAC security standard [26]. A convenient feature of the X-GTRBAC is the XML-format which not only allows it to be integrated within

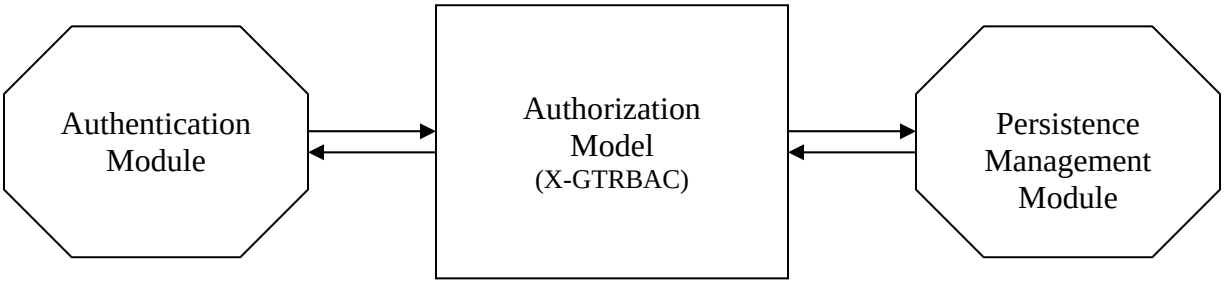

Figure 1: The design methodology for a unified distributed authentication and authorization system

Web-based applications but also makes the framework extensible. Therefore, plugging the new components into the framework does not require revisiting the complete specification; in fact it can be done in a modular fashion. In the remainder of this section, we discuss the X-Grammar specification for the persistence management and authentication modules that interface with X-GTRBAC (as shown in Figure 1) to extend it to provide support for federated identity and privilege management. The next section explores the software architecture of the system, and Appendix B presents an execution scenario of the prototype implementation of our model.

Table 1. Salient Features of X-GTRBAC

Element Type	Element Name	Purpose
RBAC Element	*XML User Sheet (XUS)*	Declares the users and their authorization credentials
	XML Role Sheet (XRS)	Declares the roles, their attributes, role hierarchy, and any separation of duty and temporal constraints associated with roles
	XML Permission Sheet (XPS)	Declares the available permissions
RBAC Assignments	*XML User-to-Role Assignment Sheet (XURAS)*	Defines the rules for assignment of users to roles; these assignments may have associated temporal constraints
	XML Permission-to-Role Assignment Sheet (XPRAS)	Defines the rules for assignment of permissions to roles; these assignments may have associated temporal constraints
RBAC Constraints	*XML Separation Of Duty Definition Sheet (XSoDDef)*	Defines the separation of duty constraints on roles
GTRBAC Constraints	*XML Temporal Constraint Definition Sheet (XTempConstDef)*	Defines the temporal constraints on role enabling and activation; also defines temporal constraints for user-to-role and permission-to-role assignments
	XML Trigger Definition Sheet (XTrigDef)	Defines context-based triggers for invocation of periodic events subject to associated constraint evaluation
Authorization Credentials	*XML Credential Type Definition Sheet (XCredTypeDef)*	Defines the available credential types

As has been outlined as one of the requirements, attention has been paid during the interface design to the fact that it should support, and not duplicate, the functionalities available in existing standards. Although many specifications are in the works, one of them has recently been hailed by the industrial community as the true enabling technology for SSO, namely the Security Assertion Markup Language (SAML) [27]. SAML provides a message exchange protocol between autonomous business entities, and is intended to be used to encode security “assertions”. The assertions are declarations of facts about an individual or business entity, much like the Attribute Certificates of X.509 PMI. An assertion, however, can also represent an authentication or authorization decision. SAML assertions can also be digitally signed. In addition, SAML supports a query/response protocol to request and send assertions. Despite all these properties, SAML is not a self-sufficient mechanism to ensure SSO as it does not provide any authentication or authorization support; it does the important task of allowing the communicating entities exchange security information in a decentralized manner but does not establish, check or revoke any information on its own. Therefore, a mechanism is needed that SAML can tie in to. Our specification provides one such mechanism, without replicating the functionality already provided by SAML. It is designed so as to accept SAML-encoded assertions as an acceptable form of credential.

However, that alone is not sufficient for our purposes- SAML assertions are inherently subject to the same name-binding problem that exists in the protocols it is designed to work with, such as Kerberos and X.509. Therefore, to satisfy the requirement of authentication for strangers, and that of anonymity and privacy, we have designed a specification that works with property-based TM credentials, as alluded to in Section 2. This requires a translation from SAML encoding to X-GTRBAC format, and vice versa, using XSLT.

We now discuss the design features of the persistence management and authentication modules in our framework. For the sake of space spacing, we do not reproduce the X-Grammar for the corresponding elements included in Appendix A. In the following, we elaborate on the noteworthy features w.r.t. to our present work on the enhanced X-GTRBAC model.

Table 2: Credential Configuration in Enhanced X-GTRBAC

#	Credential Type	X-GTRBAC Instance	Meaning	Applicable Scenario
1.	Identity-based	<User user_id =“john” > <UserName>John D</UserName> <CredType cred_type_id = “login” type_name = “Login”> <Header>… </Header> <CredExpr mode = ‘identity’> <passwd>temppass</passwd> </CredExpr> </CredType> </User>	The user with user_id john has the Login credential. The use of user_id in this credential is mandatory. The id is derived from the key; the key information is contained in the Header element and is used to authenticate the user together with the password.	This is an example of strong authentication with a key and password used by most enterprises; the user is identified using a login id (mapped to a key) and a password. All users presenting a valid credential MUST exist in the target system and are authenticated into an appropriate role.
2.	Capability-based	<User user_id =“any” > <UserName/> <CredType cred_type_id = “SysEngr” type_name = “SystemEngineer”> <Header>… </Header> <CredExpr mode = ‘capability’> <Domain>Engg</Domain> <IP>128.10.*.*</IP> <System>UNIX</System> </CredExpr> </CredType> </User>	Any user may have the credential SystemEngineer. The user_id “any” is a RESERVED word. The user authentication is based on the key information in the Header together with the attributes in the credential expression reflecting the capabilities. This credential may also be delegated as it is not bound to a user identity.	This is an example of inter-enterprise privilege management where authorization decisions can be based on capabilities of the user, and delegation of credentials may also be frequently required between different enterprises. The capabilities expressed in the credential MUST exist on the target system.
3.	Property-based	<User user_id =“any” > <UserName/> <CredType cred_type_id = “cust” type_name = “Customer”> <Header>… </Header> <CredExpr mode = ‘property’> <SSN>111-22-3333</SSN> <DLN>0991-09-0991</DLN> <DOB>05-21-78</DOB> </CredExpr> </CredType> </User>	Any user may have the credential Customer. The user_id “any” is a RESERVED word. The user authentication is based on the key information in the Header together with the attributes in the credential expression reflecting the properties. This credential may also be delegated as it is not bound to a user identity.	This is an example of Web-based privilege management in open systems where authorization decisions can be based on properties of unknown users without regards to specific capabilities on the target system; delegation of credentials is also an essential feature in this environment. NO PRIOR KNOWLEDGE of user identities or capabilities is assumed.

(i) TM Credential Configuration: Of particular interest is the configuration of TM credentials in different modes, namely identity-, capability-, or property-based, depending on the requirements of the application. All these modes are defined using the <!-- Credential Type Definition>. The Credential Expression sub-element has an attribute "mode" that allows one to specify the mode of credential configuration. The Header sub-element provides support for strong authentication, and the Attribute List sub-element can comprise of generic attributes defining identity, capability or property of the credential holder. This feature is particularly useful for backward compatibility with existing technologies. We give examples of TM credential configuration in these three modes in Table 2, along with suitable application scenarios involving the use of these credential types. We note that the credential configuration in capability or property-based modes allows authentication for unknown users since identity is not assumed to be known. If a user name is not provided in the credential, the key information in the Header element is used during role assignment. In the case of property-based credentials, the system also supports trust establishment while maintaining anonymity and privacy by requesting on-demand credentials until sufficient privilege level is determined according to the security policy. The sufficient privilege level in our context means that all role assignment conditions are satisfied in terms of possession of the desired properties. Integration with mechanisms such as SAML allows this on-demand credential collection to seamlessly occur, whereby the desired properties of the credential holder are verified by the respective issuers. The key difference in the capability and property-based credential types is that the attributes in the credential expression for the former are all specific to a particular enterprise environment, and comprise a set of capabilities known to exist in the system. On the other hand, the attributes in the credential expression for the latter type are not all assumed to be known in advance, and attributes can be acquired and supplied on demand to establish trust level of strangers in unknown environments using generic properties thereof, such as social security number or driver's license number. This decentralized control also results in a directory-less solution whereby no name-based directory lookup is needed.

Table 3: Constraint Specification in Enhanced X-GTRBAC*

#	Constraint type	X-GTRBAC Instance	Meaning	Applicable Scenario
1.	Role Delegation	<XRS xrs_id="xrsCust"> <Role role_id="rCust" role_name="Customer"> <Junior>Guest </Junior> <DelegationConstraint> <DelegationCondition d_expr_id="OneWeek"/> </DelegationConstraint> </Role> </XRS>	The role Customer can only be delegated if the delegation constraint is satisfied. The delegation condition on the role refers to a duration expression which imposes a restriction on the time period of the delegation.	This is an example of requiring the use of restrictions in privilege delegation. The restricted delegation applies to all junior roles of this role, and is enforced through the role hierarchy.
2.	Role Assignment	<URA ura_id="uraCust" role_name="Customer"> <AssignUser user_id="any"> <AssignConstraint> <AssignCondition cred_type="Customer"> </AssignConstraint> </AssignUser> </URA>	The role Customer can only be assigned to a user who possesses the credential Customer. This refers to the property-based credential (#3) in Table 2.	This is an example of requiring the use of property-based credential for assignment of unknown users to an appropriate role.

* This represents only a subset of access constraints in X-GTRBAC. For complete specification, see [12].

(ii) Delegation: Also of interest is the mechanism that enables delegation of authority to achieve the decentralization support in our framework. This requirement is captured naturally and elegantly through the use of role hierarchy in our RBAC mechanism: a senior role can set the delegation rights for its junior roles in its role definition by specifying an optional Delegation Constraint sub-element within <!-- Role Definition>. The delegation constraint may be used to restrict the interval, period or duration of the delegation using the periodic time expression of X-GTRBAC (See Table 1). The absence of a delegation constraint means unrestricted delegation rights; otherwise, the provided conditions need be satisfied in order for delegation to occur. These conditions are evaluated and enforced using the same predicate-based mechanism already in place for handling access constraints in X-GTRBAC. We illustrate the use of this constraint specification mechanism in Table 3 by listing an instance each of delegation and assignment constraint using the credential type from Table 2.

While using role-hierarchy is a particularly neat mechanism for handling organizational delegation, it is sometimes also desirable to support inter-organization delegation, for e.g., delegating one's privileges to a Web service for using them on one's behalf. Such delegation is possible by assigning the service an appropriate external role outside of the role hierarchy. The delegation can then occur from the given role to the external role by maintaining a mapping, for which we use the Linked Role sub-element which links to the role definition of the corresponding external role. The "type" attribute indicates the direction of delegation. Note that resolving a delegation chain would require a reverse lookup of key information corresponding to each instance of a linked role.

(iii) Digital Signatures: An effective SSO solution depends on the persistence of the authentication and authorization assertions across enterprise domains. Toward this end, the Header element also includes support for digital signatures. The support for digital signatures in SAML allows signed assertions to be exchanged between all SAML-compliant entities.

4. Software Architecture

In this section, we present the software architecture of our federated identity and privilege management solution. It is depicted in Figure 2.

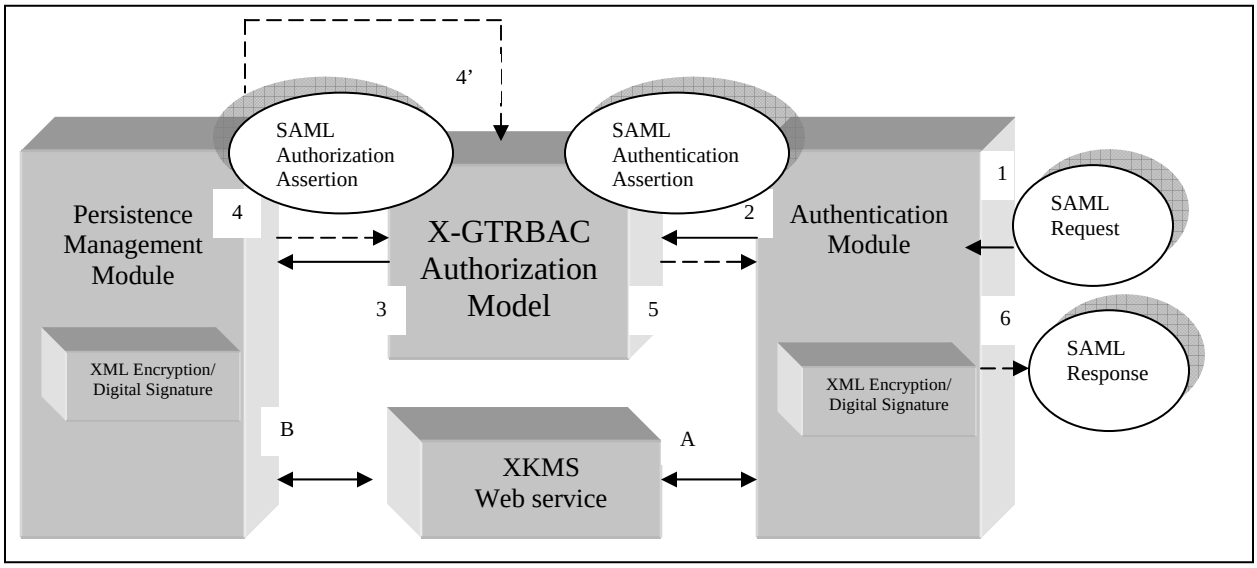

Figure 2: The software architecture for a federated identity and privilege management solution

Because of our motivation for integration with open standards, we support SAML encoding for representing authentication and authorization credentials. In the discussion henceforth, we shall refer to a signed credential as a "token", much like a Kerberos ticket, allowing the credential holder to reuse it without subsequent revalidation. Additionally, SAML encoding is also supported for the query/response protocol for credential collection. As discussed above, this particular feature allows trust establishment between strangers and also preserves anonymity and privacy by controlling disclosure of sensitive credentials according to the security policy. We outsource the certificate management to the well-known XML Key Management Specification (XKMS) [28]. XKMS is a Web-based service that can be invoked from a client application, and supports PKI-based key generation (at either client or server), registration, revocation, and verification. SOAP binding is used for message exchange. XML Encryption and XML Digital Signature standards are used to provide message confidentiality and authenticity, respectively. The end-to-end communication is assumed to be secured using mechanisms such as SSL/TLS.

The following scenario highlights the salient features of the system architecture (the step numbers correspond to the numbered arrows in Figure 2):

Step 1: User enters his login id and requests access to a resource. The login id may either be the user's public key or an identifier that uniquely maps to the public key. Such an identifier may be generated and mapping maintained by a dedicated software routine, or it may also be done through the use of hardware (such as smart cards). This arrangement is nevertheless desirable as users cannot be expected to enter difficult-to-remember public key values at the login console. The access request along with the login information is sent to the authentication module as a SAML request with an embedded authentication query.

Step 2: The authentication module evaluates the information in the SAML request (using either XKMS or the local server) and appropriately issues a SAML authentication assertion. In our research prototype, the authentication module itself acts as a proxy for the XKMS Web service for issuing SAML-compliant authentication assertion. The authentication assertion is appended to the security header in the SOAP message. Attribute assertions may similarly be obtained. In case the request goes to XKMS, the authentication module and XKMS can also communicate using SAML.

Step 3: Based on SAML authentication and attribute assertions, the X-GTRBAC module assigns a role membership to the requestor according to the available information. This step requires a translation from the SAML assertions into X-GTRBAC credential format which is used for user-to-role mapping. The authorizations of the user are then determined based on the corresponding role-permission mapping. Additional attribute assertions may also be obtained during this process if anonymity and privacy considerations do not allow all attributes to be declared upfront in step 2. This can be achieved by using trust negotiation mechanisms [29] to allow gradual disclosure of sensitive attributes. Once sufficient level of trust has been established, the authorization decision is captured as an X-GTRBAC credential with the holder (identified by the public key) as the role name and the attributes as the role permissions.

Step 4: To enable SSO, the X-GTRBAC module communicates the authorization credential to the persistence management module, which digitally signs it and returns an authorization token in the form of a SAML assertion. This token can subsequently be used by the requestor to access resource without going through an authentication process (step 4').

The steps A and B in Figure 2 represent the communication between the system modules and the XKMS Web service, and may be invoked as necessary during the communication. For instance, step A could be carried out by the authentication module before the start of the communication to generate and register keys, and later on to verify the same. Similarly, step B could be carried out by the persistence management module to verify the digital signatures of an authorization assertion received by the X-

GTRBAC system. Additionally, there is the option to return the SAML response to a SAML request back to the requestor (step 6). This is needed in situations when the request is initiated from an intermediary wishing to obtain assertions about the end user. Appendix B illustrates an execution scenario during prototype testing of this architecture using the policy instances from Tables 2 and 3 in the previous section.

Conclusion

This paper presented a federated identity and privilege management solution for open systems. Among the primary motivations of this work was to overcome the shortcomings of traditional distributed authentication and authorization schemes, and to develop an access management framework enabling decentralized SSO functionality across multiple enterprise domains. Our framework employs X-GTRBAC as the authorization model, and hence supports fine-grained attribute-based access control. An authentication module is integrated into X-GTRBAC for strong authentication. SSO is achieved through a privilege management mechanism integrated into X-GTRBAC for issuing signed authorization assertions. The use of property-based credentials presents a scalable alternative to name-based and capability-based approaches. It not only allows SSO to be decentralized, but also help with anonymity and privacy since it allows incremental trust establishment to occur. A particularly convenient feature of our approach is its integration with SAML, a current standard aimed at enabling SSO. To the best of our knowledge, ours is the first approach integrating two security standards, namely RBAC and SAML, toward designing an access management framework for open systems. Overall, our grammar specification provides support for federated identity and privilege management while meeting the requirements outlined in the paper. However, we believe that this set of requirements is not exhaustive. We have only presented an improved mechanism; it is not necessarily ideal yet. Among some challenges we see presently are integration with existing directory schemes to support property-based credentials, maintaining some state information for anonymous users to ensure proper accountability, and handling delegation in the presence of autonomous linked roles such as Web services, which requires ontology-mediated resolution of external roles along a delegation chain. A prototype system of our current model has been implemented and preliminarily tested. We intend to report detailed implementation experiences in some future work. We also plan to integrate our system with a trust negotiation system like Trust-X [29] and to extend it with privacy enhancing techniques [30].

References

[1] D. Buell, R. Sandhu, “Guest Editors' Introduction: Identity Management”, IEEE Internet Computing, Nov/Dec2003.

[2] Microsoft .NET Passport http://www.passport.net/Consumer/Default.asp?lc=1033

[3] R. M. Needham and M. D. Schroeder, “Using encryption for authentication in large networks of computers," Communications of the ACM, vol. 21, no. 12, pp. 993{999, 1978.

[4] Kerberos: The Network Authentication Protocol, http://web.mit.edu/kerberos/www/

[5] R. M. Needham and A. H. Herbert, “The Cambridge Distributed Computing System”. Addison Wesley, Jan. 1982. ISBN 0-20114-092-6.

[6] A. Wulf, E. S. Cohen, W. M. Corwin, A. K. Jones, R. Levin, C. Pierson, and F. J. Pollack, “HYDRA: The kernel of a multiprocessor operating system," Communications of the ACM, vol. 17, pp. 337-345, June 1974.

[7] S. J. Mullender, C. van Rossum, A. S. Tanenbaum, R. van Renesse, and H. van Stavern, “Amoeba: a distributed operating system for the 1990s.," IEEE Computer, vol. 23, pp. 44-53, May 1990.

[8] M. Myers, C. Adams, D. Solo, and D. Kemp, “Internet X.509 certi_cate request message format," RFC 2511, Internet Engineering Task Force, Mar. 1999. See http://www.ietf.org/rfc/rfc2511.txt.

[9] C. M. Ellison, “SPKI requirements," RFC 2692, Internet Engineering Task Force Draft IETF, Sept. 1999. See http://www.ietf.org/rfc/rfc2692.txt.

[10] M. Blaze, J. Feigenbaum, and A. D. Keromytis, “KeyNote: Trust management for public-key infrastructures," in Security Protocols International Workshop, Springer LNCS, no. 1550, pp. 59-63, 1998.

[11] ITU-T (Telecommunication Standardization Sector, International Telecommunication Union), Geneva, Switzerland, ITU-T Recommendation X.509: The Directory: Public-Key and Attribute Certificate Frameworks, 2000.
[12] R. Bhatti, "X-GTRBAC: An XML-based Policy Specification Framework and Architecture for Enterprise-Wide Access Control", Masters thesis, Purdue University, May 2003. Available as CERIAS tech. report 2003-27.
[13] J. Bacon, K. Moody, and W. Yao, "Access control and trust in the use of widely distributed services", In Middleware 2001, volume LNCS 2218, pages 300{315. Springer-Verlag, November 2001.
[14] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor, and Y. Ravid, "Access control meets public key infrastructure, or: Assigning roles to strangers", In Proceedings of the 2000 IEEE Symposium on Security and Privacy, pp. 2–14, 2000. IEEE Press.
[15] Ninghui Li, John C. Mitchell, and William H. Winsborough, "Design of a role-based trust management framework", In Proceedings of the 2002 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, May 2002.
[16] W. Yao, "Trust Management for Widely Distributed Systems", PhD Thesis, University of Cambridge.
[17] Gombás Gábor, "Evaluation of Distributed Authentication, Authorization and Directory Services",http://www.caesar.elte.hu/eltenet/projects/demogrid/demogrid-report-1/dg-rep-1-sec-eval.pdf
[18] J. B. D. Joshi, W. G. Aref, A. Ghafoor and E. H. Spafford, "Security Models for Web-based Applications", *Communications of the ACM*, 44, 2 (Feb. 2001), pages 38-72.
[19] R. Bhatti, J. B. D. Joshi, E. Bertino, A. Ghafoor, "XML based Specification for Web- Services Document Security", IEEE Computer, April 2004.
[20] D. Chadwick, A. Otenko, "The PERMIS X.509 role based privilege management infrastructure", In Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies, June 2002.
[21] J. Park, R. Sandhu, "RBAC on the Web by smart certificates", In Proceedings of the Fourth ACM Symposium on Access Control Models and Technologies, October 1999.
[22] Extensible Access Control Markup Language (XACML) http://xml.coverpages.org/xacml.html
[23] David P. Kormann and Aviel D. Rubin, "Risks of the Passport Single Signon Protocol", Computer Networks, Elsevier Science Press, volume 33, pages 51-58, 2000.
[24] R. Bhatti, E. Bertino, A. Ghafoor, "A Trust based Context-Aware Access Control Model for Web Services", In proceedings of The Third International Conference on Web Services, San Diego, July 6-9, 2004.
[25] R. Bhatti, J. B. D. Joshi, E. Bertino, A. Ghafoor, "X-GTRBAC Admin: A Decentralized Administration Model for Enterprise Wide Access Control", In proceedings of 9th ACM Symposium on Access Control Models and Technologies (SACMAT04), 2-4 June 2004
[26] Role Based Access Control (RBAC): ANSI Security Standard http://xml.coverpages.org/ni2004-04-05-a.html
[27] Security Assertion Markup Language (SAML) http://xml.coverpages.org/saml.html
[28] XML Key Management Specification http://www.w3.org/TR/xkms/
[29] E.Bertino, E.Ferrari, A.C. Squcciarini, "A Peer-to-Peer Framework for Trust Establishment", To appear in IEEE Transactions on Knowledge and Data Engineering, 2004.
[30] Proceedings of Fourth Workshop on Privacy Enhancing Technologies (PET 2004), Toronto (Canada), May 26-28, 2004.

APPENDIX A

X-GTRBAC Grammar
[Basic Definitions]

```
<!-- Policy Definition> ::=<Policy policy_id =(id)>
   <PolicyName> (name) </PolicyName>
   <!-- XML User Sheet>
   <!-- XML Role Sheet>
   <!-- XML Permission Sheet>
   <!-- XML User-Role Assignment>
   <!-- XML Permission-Role Assignment>
   [<!-- Local Policy Definitions>]
   [<!-- Policy Relationship Definitions>]
</Policy>
<!-- XML User Sheet>    ::=<XUS [xus_id = (id) ]>
      [<!-- Definitions of Credential Types>]
      <!-- User Definitions>
</XUS>
<!-- Definitions of Credential Types>
   ::=  <XCredType [xctd_id = (id) ] >
      [<!-- Credential Type Definition>]+
</XCredType>
<!-- Credential Type Definition>
::=     <CredType cred_type_id = (id)
type_name= (type name) >
   <!-- Attribute List>
</CredType >
<!-- Attribute List>  ::= <AttributeList>
      [<!-- Attribute Definition>]+
</AttributeList>
<!-- Attribute Definition> :: <Attribute>
   <AttributeName usage = "mand | opt"
      type = (type)> (name) </AttributeName >
</Attribute>
<!-- User Definitions >    ::=<Users>
       [<!-- User Definition>]+
</Users>
<!-- User Definition> ::=      <User user_id = (id)>
     <UserName>[(name)]</UserName>
     <!—CredType>
     <MaxRoles>(number)</MaxRoles>
   </User>
<!—CredType > ::=  <CredType cred_type_id = (id)
                   type_name= (type name) >
   [<!—Header>]
   <!-- Credential Expression>
</CredType>
<!-- Credential Expression > ::= <CredExpr mode=
(identity | capability | property)>
   <!-- AttributeValuePairs>
   <!-- DomainSet>
</CredExpr>
<!-- AttributeValuePairs> ::= [<(attribute name)> ( attribute
value) </(attribute name)>] +
<!-- XML Role Sheet>    ::=<XRS [xrs_id = (id) ]>
   [<!-- Role Definition>]+
</XRS>
<!-- Role Definition>      ::=<Role role_id = (id)
        role_name = (role name)>
   [<!-- Attributes>]
   <!-- DomainSet>
   [<!—(En|Dis)abling Constraint>]
   [<!—[De]Activation Constraint>]
   (<SSDRoleSetID> (id) </SSDRoleSetID>)*
   (<DSDRoleSetID> (id) </DSDRoleSetID>)*
   [<Junior> (name) </Junior>]
   [<Senior> (name) </Senior>]
   [<LinkedRole type=(delegator |
delegatee)>(name)</LinkedRole>]
   [<!—Delegation Constraint>]
   [<Cardinality> (number) </Cardinality>]
</Role>
<!-- Attributes>   ::=     <Attributes>
<!-- AttributeValuePairs>
<!-- Separation of Duty Definitions>
::=    <XSoDDef [xsod_id = (id) ]>
        [<!—SSDRoleSets>]
        [<!—DSDRoleSets>]
      </XSoDDef>
<!-- SSDRoleSets > ::=     <SSDRoleSets>
        [<!—SSDRoleSet>]+
        </SSDRoleSets>
<!—SSDRoleSet> ::=    <SSDRoleSet>
          [<SSDRole ssd_role_set_id =(id)
            ssd_cardinality = (number)>
          (role name)
        </SSDRole>]+
   </SSDRoleSet>
<!-- DomainSet>  ::=     <DomainSet>
                    [<!—DomainID>]+
   </DomainSet>
<!-- DomainID>::=     <DomainID>(id)</DomainID>
<!-- DSDRoleSets >  ::=     <DSDRoleSets>
   [<!—DSDRoleSet>]+
   </DSDRoleSets>
<!—DSDRoleSet>::=    <DSDRoleSet>
     [<DSDRole dsd_role_set_id =(id)
         dsd_cardinality = (number)>
         (role name)
   </DSDRole>]+
   </DSDRoleSet>
<!-- XML Permission Sheet>::=<XPS [xps_id = (id) ]>
   [<!-- Permission Definition>]+
</XPS>
<!-- Permission Definition>  ::=
<Permission perm_id = id [prop= (prop op)] >
<Object type= (type name) id= (id)/>
<Operation> (access op) </Operation>
<!-- DomainSet>
</Permission>
<!-- XML User-Role Assignment Sheet>::=
<XURAS [xuras_id = (id) ]>
   [<!-- User-role Assignment>]+
</XURAS>
<!-- User-role Assignment>::=
<URA        ura_id=(id)        role_name=(name)>
   <AssignUsers>
      [< !—Assign User>]+
   </AssignUsers>
</URA>
<!—[De]Assign User >                ::=
   <[De]AssignUser user_id=(id)>
   <!—[De]Assign Constraint >
   </[De]AssignUser>
<!-- XML Permission-Role Assignment Sheet>::=
<XPRAS [xpras_id = (id) ]>
   [<!-- Permission-Role Assignment>]+
</XPRAS>
```

```
<!-- Permission-Role Assignment>::=
  <PRA pra_id=(id)  role_name=(name)>
  <AssignPermissions>
      [<!—Assign Permission>]+
</AssignPermissions>
</PRA>
<    !—[De]Assign     Permission>                    ::=
<[De]AssignPermission perm_id=(id)>
<!—[De]Assign Constraint >
</[De]AssignPermission>
<!—[De]Assign Constraint> ::=
  <[De]AssignConstraint[op =AND|OR|NOT|XOR)]>
    // opcode defaults to AND if none specified
  [<!—[De] Assign Condition>]+
</[De]AssignConstraint>
<!—[De]Assign Condition>            ::=
<[De]AssignCondition cred_type=”type_name”
   [pt_expr_id=(id) | d_expr_id=(id)] >
   [<!-- Logical Expression>]
</[De]AssignCondition>
<!—(En|Dis)abling Constraint>          ::=
  <(En|Dis)abConstraint[op = (AND|OR|NOT)]>
     // opcode defaults to AND if none specified
     [<!-- (En|Dis)abling Condition>]+
  </(En|Dis)abConstraint>
<!—(En|Dis)abling Condition>          ::=
  <(En|Dis)abCondition [pt_expr_id=(id) |
                        d_expr_id=(id)] >
 [<!-- Logical Expression>]
  </(En|Dis)abCondition>
<!—[De]Activation Constraint> ::=
  <[De] ActivConstraint[op = (AND|OR|NOT)]>
   // opcode defaults to AND if none specified
     [<!—[De]ActivationCondition>]+
  </[De]ActivConstraint>
<!—[De]Activation Condition>          ::=
  <[De]ActivCondition [d_expr_id=(id)]>
   <!-- Logical Expression>]
  </[De]ActivCondition >
<!-- Logical Expression> ::=
<LogicalExpr [op = (AND|OR|NOT)]>
  // opcode defaults to AND if none specified
  [<!-- Predicate>]+
</LogicalExpr>
<!-- Predicate>           ::=        <Predicate>
   { <Operator> (gt|lt|eq|neq) </Operator>
     <NameParam>(name)</NameParam>
     <ValueParam>(value)</ValueParam> }
  | < !--LogicalExpression>
</Predicate>
```

[Temporal Definitions]

```
<!-- Definitions of Temporal Constraints>::=
  <XTempConstDef [xtcd_id = (id) ]>
     [<!—Interval Expression>]
     [<!-- Periodic Time Expression>]
    [<!-- Duration Expression>]
</XTempConstDef>
<!-- Periodic Time Expression>          ::=
  <PeriodicTimeExpr pt_expr_id = (id)
  <!-- Start Time Expression>
  </PeriodicTimeExpr>
<!—Interval Expression> ::=
<IntervalExpr i_expr_id = (id)>
<begin> (date)</begin>
 <end>(date)</end>
</IntervalExpr>
<!-- Start Time Expression>   ::=       <StartTimeExpr
[pt_id_ref = (pt_id)]>
  [<Year>(all|odd|even) /<Year>]
  [<!--MonthSet>]
  [<!--WeekSet>]
  [<!--DaySet>]
</StartTimeExpr>
<!--MonthSet>    ::=<MonthSet>
  (<Month>(1|..|12)</Month>)1-12
    (represents # of months from the start of current Year)
</MonthSet >
<!--WeekSet>              ::=         <WeekSet>
  (<Week>(1|..|4)</Week>)1-4
  (represents # of weeks from the start of current Month)
</WeekSet >
<!--DaySet>               ::=         <DaySet>
  (<Day>(1|..|7)</Day>)1-7
  (represents # of days from the start of current Week)
</DaySet >
<!-- Duration Expression>  ::=
<DurationExpr d_expr_id = (id)>
  <cal>(Years|Months|Weeks|Days)</cal>
  <len> (number)</len>
</DurationExpr>
```

[TM Credential Definitions]

```
<!--Header>       ::=  <Header>
    <!--  Principal >
    <!--  Issuer >
    <!--  Validity>
     [<!--  Digital Signature >]
    </Header>
<!-- Issuer>        ::=  <Issuer>
  <!-- Principal>
  </Issuer>
<!-- Principal>::= <Principal short_name = (ID)>
      {<PublicKey>(Hash ID)</PublicKey> |
       <NameToken>(String)</NameToken>}
</Principal>
<!-- Validity>       ::= <Validity>
  <IssueTime>(xs:dateTime)</IssueTime>
  [<NotBefore>(xs:dateTime)</NotBefore>]
  [<NotAfter>(xs:dateTime)</NotAfter>]
  </Validity>
<!--  Digital Signature >  ::= <DSig>
  (ds:Signature) </DSig>
<!--  Hash ID >   ::=  xs:base64Binary
<!—Delegation Constraint>   ::=
<DelegationConstraint [op = (AND|OR|NOT)]>
 // opcode defaults to AND if none specified
[<!-- Delegation Condition>]+
<!—Delegation Condition>       ::=
  <DelegationCondition [pt_expr_id=(id) |
                        d_expr_id=(id)] >
    [<!-- Logical Expression>]
</DelegationCondition>
```

APPENDIX B

Prototype Implementation
[An Execution Scenario]

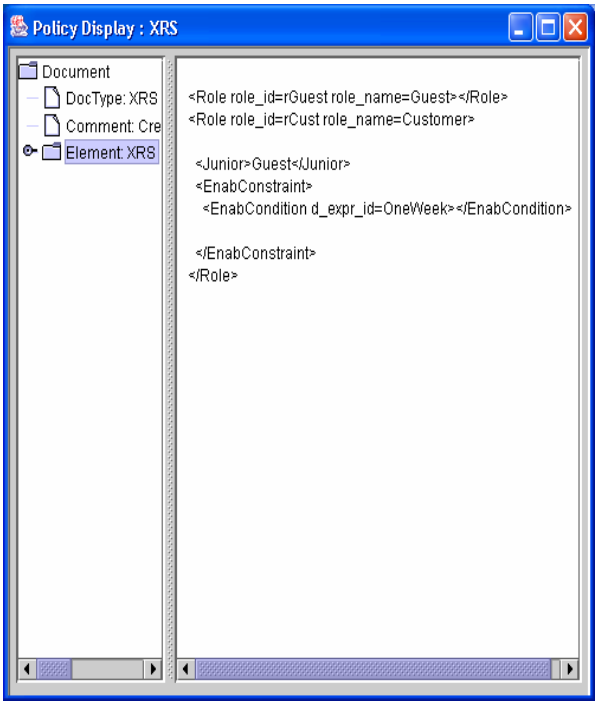

Figure B.1: Policy display of the XML Role Sheet showing the `Customer` role information

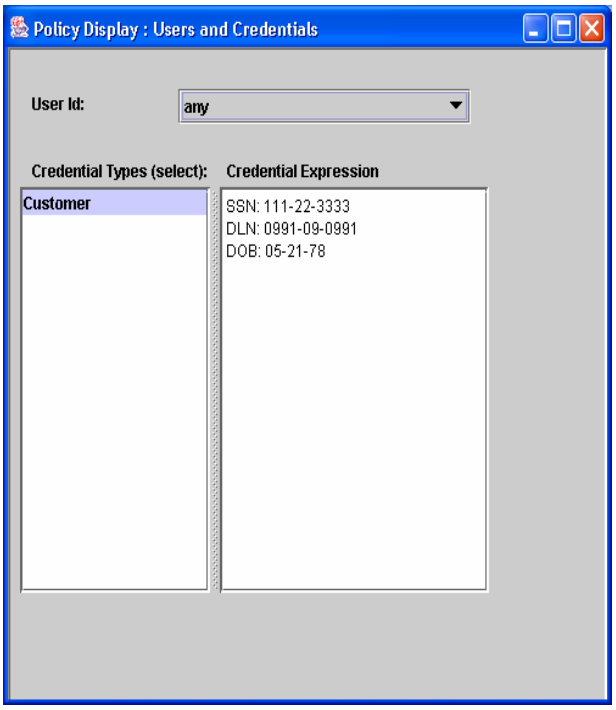

Figure B.2: Policy display of the XML User Sheet showing attributes for a `Customer` credential.

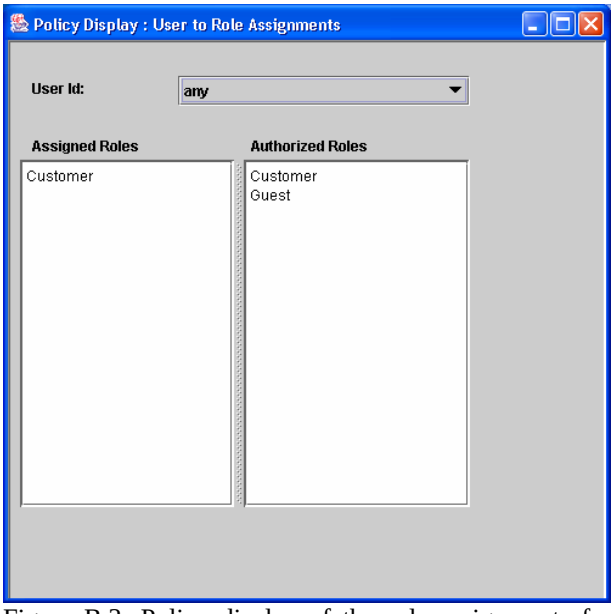

Figure B.3: Policy display of the role assignments for the "any" user. Note that the user has been authenticated into the `Customer` role based on the `Customer` credential of Figure B.2.

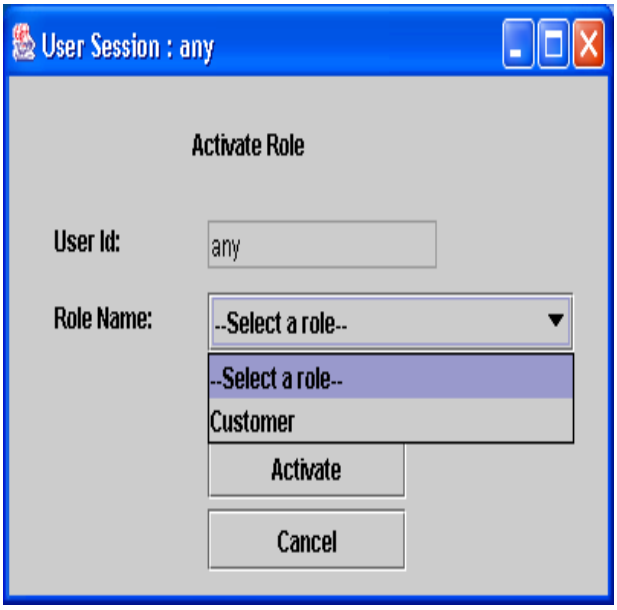

Figure B.4: Screen capture showing the initiation of a user session for the "any" user. The user can select from the assigned roles in the list and obtain the corresponding authorizations.