Access Control in Collaborative Systems

WILLIAM TOLONE, GAIL-JOON AHN, AND TANUSREE PAI

University of North Carolina at Charlotte

AND

SENG-PHIL HONG

Information and Communications University

Balancing the competing goals of collaboration and security is a difficult, multidimensional problem. Collaborative systems often focus on building useful connections among people, tools, and information while security seeks to ensure the availability, confidentiality, and integrity of these same elements. In this article, we focus on one important dimension of this problem—access control. The article examines existing access control models as applied to collaboration, highlighting not only the benefits, but also the weaknesses of these models.

Categories and Subject Descriptors: K.6.5 [**Management of Computing and Information System**]: Security and Protection

General Terms: Management, Security

Additional Key Words and Phrases: Access control, collaboration, security models

1. INTRODUCTION

Collaborative systems, groupware, or multi-user applications allow groups of users to communicate and cooperate on common tasks. Example systems include a wide range of applications such as audio/video conferencing, collaborative document sharing/editing, distance learning, workflow management systems, and so on. All of these systems contain information and resources with different degrees of sensitivity. The applications deployed in such systems create, manipulate, and provide access to a variety of protected information and resources.

Balancing the competing goals of collaboration and security is difficult because interaction in collaborative systems is targeted towards making people,

The work of Gail-J. Ahn was partially supported by the grants from National Science Foundation (NSF-IIS-0242393) and Department of Energy Early Career Principal Investigator Award (DE-FG02-03ER25565).

Authors' address: Department of Software and Information Systems, College of Information Technology, University of North Carolina at Charlotte, 9201 University City Blvd., Charlotte, NC 28223-0001; email: {wjtolone,gahn,tpai}@uncc.edu; url: www.sis.uncc.edu/LIISP; S.-P. Hong, Information and Communications University, Taejon, Korea; email: philhong@icu.ac.kr.

Corresponding author: Dr. Gail-J. Ahn, gahn@uncc.edu.

information, and resources available to all who need it, whereas information security seeks to ensure the availability, confidentiality, and integrity of these elements while providing it only to those with proper authorization. Protection of contextual information and resources in such systems therefore entails addressing several requirements not raised by traditional single-user environments, due in part to the unpredictability of users and the unexpected manners in which users and applications interact in collaborative sessions.

Among the several areas of security under consideration for collaborative environments, authorization or access control is particularly important because such systems may offer open access to local desktops or networked resources, for example, H.323 and T.120 conferencing tools need to support text-based chat, audio/ videoconferencing, shared whiteboard, and application and screen sharing. Users need a mechanism not only for identifying collaborators through proper authentication, but to manage which files, applications, portions of a system, and so forth. they can access during a collaboration session. In this article, we provide a comprehensive study of authorization mechanisms for collaborative environments examining both the merits and weaknesses of each approach. Based on this study, we outline best practices in access control, while addressing the unique authorization requirements for collaboration.

The rest of this article is organized as follows. Section 2 discusses access control requirements for collaboration as documented from existing research. Section 3 examines existing access control models as applied to existing collaborative environments in light of these requirements, highlighting not only the benefits, but, more importantly, the weaknesses of these models. In section 4, we assess these models based on criteria drawn from our study. Section 5 discusses lessons learned from our experiment and concludes the article.

2. ACCESS CONTROL REQUIREMENTS FOR COLLABORATION

Access control models are used to decide on the ways in which the availability of resources in a system are managed and collective decisions of the nature of the environment are expressed. Several groups [Edwards 1996; Jaeger and Prakash 1996; Ferraiolo and Barkley 1997; Bullock 1998] have studied the requirements for access control in collaborative environments. We summarize these requirements as follows.

—Access control must be applied and enforced at a distributed platform level.

—Access control models should be generic and enable access rights to be configured to meet the needs of a wide variety of cooperative tasks and enterprise models. That is, such models should be expressive enough to specify access rights efficiently based on varied information (e.g., roles, context).

—Access control for collaboration requires greater scalability in terms of the quantity of operations than tradition single user models because the number of shared operations is much richer in collaborative environments compared to traditional single user systems.

—Access control models must be able to protect information and resources of any type and at varying levels of granularity. That is, they must have the ability to provide strong protection for shared environments and objects of various types as well as allow fine-grained control of access to individual objects and their attributes.

—Access control models must facilitate transparent access for authorized users and strong exclusion of unauthorized users in a flexible manner that does not constrain collaboration.

—Access control models must allow high-level specification of access rights, thereby better managing the increased complexity that collaboration introduces.

—Access control models for collaboration must be dynamic, that is, it should be possible to specify and change policies at runtime depending on the environment or collaboration dynamics.
—Performance and resource costs should be kept within acceptable bounds.

These features are desirable for collaboration, but we still need to consider other access control requirements in collaborative environments. For example, meta access control or access administration is also a relevant requirement. Meta access control can either be incorporated within the basic model for access control or provided through a separate model. Requirements for meta access control in collaborative environments have been studied [Dewan and Shen 1998] and include support for fine-grained protection, assignment of administrators, joint and multiple ownership issues, and the delegation and revocation of access rights. It is futile to try to enumerate all interesting and practically useful access control requirements because there are too many possibilities and variations. Instead, we should pursue intuitively simple yet rigorous access control models for specifying and enforcing access control requirements. This study attempts to identify the weaknesses and strengths of existing access control models in the context of collaborative environments so that we can eventually propose necessary criteria for such models.

3. ACCESS CONTROL MODELS FOR COLLABORATION

In this section, we examine existing access control models for collaborative environments. As part of this examination, we present an overview of the principles and merits of each approach as well as identify potential shortcomings.

3.1. Access Matrix Model

The subject-object distinction is basic to access control [Sandhu and Samarati 1994]. Subjects initiate actions or operations on objects. These actions are permitted or denied based on the authorizations specified in the system. Authorization is expressed in terms of access rights or access modes.

	File 1	File 2	File 3	File 4
John	Own R W		Own R W	
Alice	R	Own R W	W	R
Bob	R W	R		Own R W

Fig. 1. Access matrix model.

The access matrix is a conceptual model which specifies the rights that each subject possesses for each object. The access matrix [Lampson 1971] provides a useful framework for describing resource protection in operating systems. This model defines three kinds of access-control entities: a) protected objects—the entities/resources which can be accessed; b) subjects—the active entities who access objects; and c) access rights which associate the subject with the protected objects by specifying the operations that subjects are allowed to perform on objects. An access matrix `A`, with rows representing subjects, columns representing objects is used to define the protection state. `A[s, o]` denotes the access rights a subject `s` has over an object `o`. The access-checking rule of the model states that a request by subject `s` for accessing object `o` is granted only if `A[s, o]` contains the requisite right. This is achieved by means of a reference monitor which is responsible for mediating all attempted operations by subjects on objects.

An example of an access matrix is shown in Figure 1, where the rights R and W denote read and write, respectively, and the other rights are as previously discussed. This matrix specifies that, for example, John is the owner of File 3 and can read and write that file, but John has no access to File 2 or File 4. Since John owns File 1, he can give Alice the R right and Bob the R and W rights, as shown in Figure 1. John can later revoke one or more of these rights at his discretion.

A common approach to implementing the access matrix is by means of access

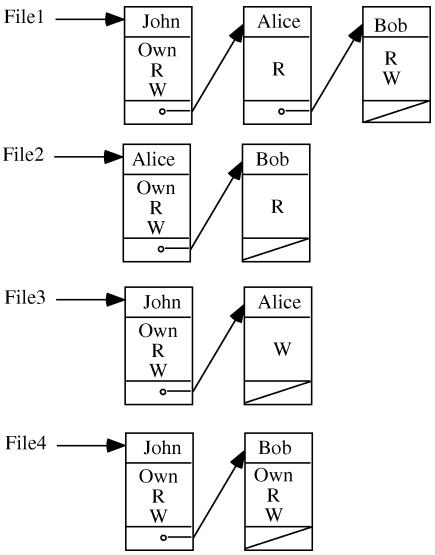

Fig. 2. Access control list.

control lists (ACLs). Each object is associated with an ACL, indicating for each subject in the system the accesses the subject is authorized to execute on the object. This approach corresponds to storing the matrix by columns. ACLs corresponding to the access matrix of Figure 1 are shown in Figure 2. Basically the access matrix column for File 1 is stored in association with File 1, and so on.

Capabilities are a dual approach to ACLs. Each subject is associated with a list, known as the capability list, indicating for each object in the system, the accesses the subject is authorized to perform on the object. This approach corresponds to storing the access matrix by rows. Figure 3 shows capability lists for the files in Figure 1.

3.1.1. Shortcomings. There are several weaknesses to the access matrix model. Some are more general, while others are magnified due specifically to collaboration requirements.

—More sophisticated access policies such as access based on competency, least-privilege, or conflict-of-interest rules are difficult to provide without access rights that are associated with a subject's credentials when performing an operation.

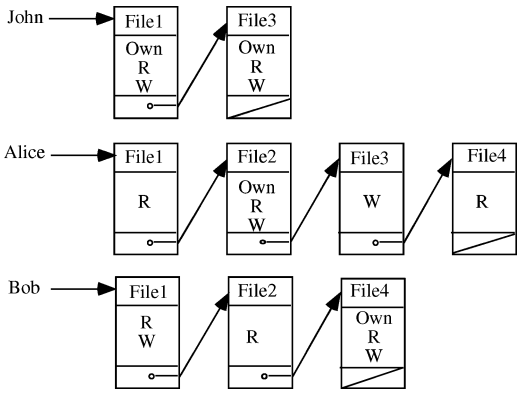

Fig. 3. Capability list.

—Users might change responsibilities at any point. ACL- and capability-based approaches lack the ability to support dynamic changes of access rights. For an object's ACL, it is easy to determine which modes of access subjects are authorized to have for that object. That is, ACLs provide for convenient access review with respect to an object. It is also easy to revoke all access rights to an object by replacing the existing ACL with an empty access mode. On the other hand, determining all the accesses that a subject has is difficult in an ACL-based system. It is necessary to examine the ACL of every object in the system to review access privileges with respect to a subject. Similarly, if all accesses of a subject need to be revoked, all ACLs must be checked one by one. In a capability list approach, it is easy to review all accesses that a subject is authorized to execute by reviewing the subject's capability list. But, determining all subjects who are allowed to access a particular object requires reviewing every capability list of each subject.

—In a collaborative or organizational workflow setup, ownership might not be at the discretion of the user, that is, the system might own resources. ACLs and C-lists inadequately address this issue. Access rights may be related to content, attribute of resources, or other contextual information. Access matrices do not account for this situation.

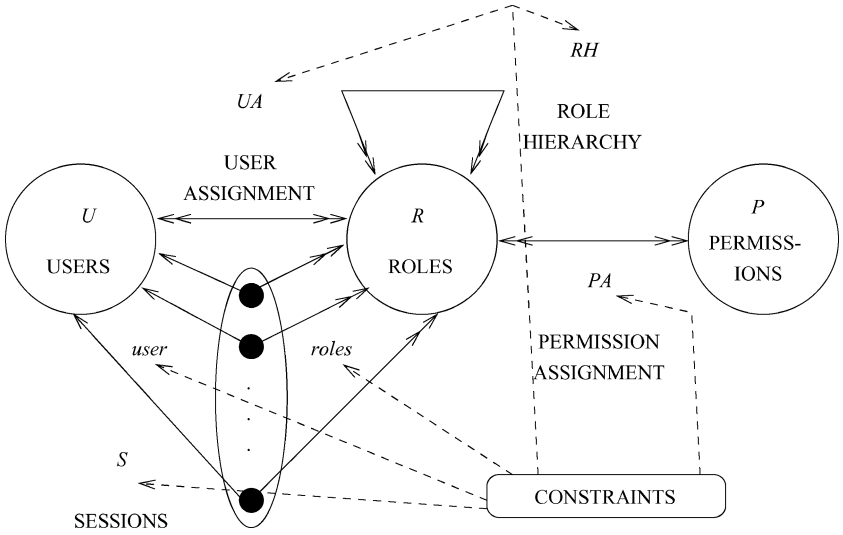

Fig. 4. Role-based access control.

3.1.2. Collaborative Frameworks. Examples of collaborative systems using ACLs are the Grove Outline Editor [Ellis et al. 1989] which supports single ownership; RTCAL [Grief and Sarin 1987] which uses a central administrator instead of owners for specifying, granting, and revoking of access rights; and Diva [Sohlenkamp and Chwelos 1994] which uses a global access policy, that is, access is granted based on the target of access and not the requestor. Examples of frameworks that use roles in conjunction with ACLs to specify access models are the Intermezzo framework [Edwards 1996] and the SUITE model proposed by Shen and Dewan [1992].

The key features in the Intermezzo collaborative framework [Edwards 1996] are data storage and replication, authentication and authorization, and awareness service. It supports single authorizer for resources and therefore has simple access administration rules. It has a policy system that uses roles to categorize groups of users with a common set of rights and introduces the notion of dynamic roles which allows for membership to be specified at runtime depending on the access request.

The access control model proposed by Shen and Dewan [1992] supports multiple roles for users and identifies a set of collaborative rights. Inheritance rules are used on the subject, object, and access rights dimensions to simplify management of roles and specification of rights. Negative rights are used to simplify the definition of policies. At the same time, positive and negative rights necessitate conflict resolution rules.

Sikkel [1997] defined an access control system for the BSCW system. This model avoids the need for conflict resolution rules by introducing negative group membership rather than negative rights on access.

3.2. Role-Based Access Control (RBAC)

The essence of RBAC [Sandhu et al. 1996] is that permissions are assigned to roles rather than to individual users. Roles are created for various job functions, and users are assigned to roles based on their qualifications and responsibilities. This way, the task of specifying user authorization is divided into two logically independent parts: one which assigns users to roles and one which assigns access rights for objects to roles as illustrated in Figure 4.

RBAC96 family of models [Sandhu et al. 1996] supports the notion of role activation within sessions, where session is a concept that is bound to a single user and

allows the user to activate the permissions of a subset of roles to which he/she belongs.

From a policy perspective, the capability within RBAC to impose constraints on user membership by assigning users to roles provides a powerful means of enforcing conflict of interest and cardinality rules for roles as they uniquely apply to a collaborative environment [Ahn and Sandhu 2000]. Users can be easily reassigned from one role to another without modifying the underlying access structure. RBAC is thus more scaleable than user-based security specifications and greatly reduces the cost and administrative overhead associated with fine-grained security administration at the level of individual users, objects, or permissions.

3.2.1. Shortcomings. While very effective and popular for traditional and collaborative systems, RBAC has several weaknesses.

—Most early implementations of RBAC determined the set of roles in use as well as the role membership early in the lifetime of a session. Changes were not well supported. The nature of such roles could be called static—they lacked flexibility and responsiveness to the environment in which they were used.

—RBAC96 supports the notion of role activation within sessions, but it does not go far enough in encompassing the overall context associated with any collaborative activity. The importance of context in the activation, deactivation, and management of permissions was identified when security systems were classified as passive security systems and active security systems. A passive security system is one that primarily serves the function of maintaining permission assignments, such as in RBAC where permissions are assigned to roles. An active security system, on the other hand, takes into account the impact of context as it emerges with progressing tasks and distinguishes task- and context-based permission activation from permission assignment.

—Traditional RBAC lacks the ability to specify a fine-grained control on individual users in certain roles and on individual object instances. For collaborative environments, it is insufficient to have role permissions based on object types. Rather, it is often the case that a user in an instance of a role might need a specific permission on an instance of an object.

—Another important issue in the RBAC model implementation is the power of constraints specification. Constraints are an important aspect of role-based access control and a powerful mechanism for laying out higher-level organizational policy. The importance of flexible constraints to support emerging applications has been recently discussed by Jaeger [1999] and Ahn and Sandhu [2000]. However, the specification of such constraints have not been discussed in the RBAC model.

3.2.2. Collaborative Frameworks. Several systems used the notion of roles to define access control groups even before the RBAC model was formally accepted. Examples of such systems are MPCAL [Grief and Sarin 1987], ICICLE [Brothers et al. 1990], SUITE [Shen and Dewan 1992], ConversationBuilder [Kaplan et al. 1992], PREP [Neuwirth et al. 1990], and WORLDS [Kang et al. 2001]. Most of these systems, however, concentrate on individuals working together.

Requirements of an RBAC system for implementing a DAC model were suggested in Jaeger and Prakash [1996]. Such a model proposes to enable users and their applications to control access at runtime. Several RBAC-based systems have made attempts to make the system as active as possible by recognizing context and or activation rules [Zhang et al. 2001, 2003; Ahn et al. 2003; Shin et al. 2002; Ahn et al. 2004].

Issues of role activation and deactivation in RBAC systems have been addressed in models such as OASIS [Yao et al. 2001] and Temporal-RBAC [Bertino et al. 2000]. OASIS uses parameters based

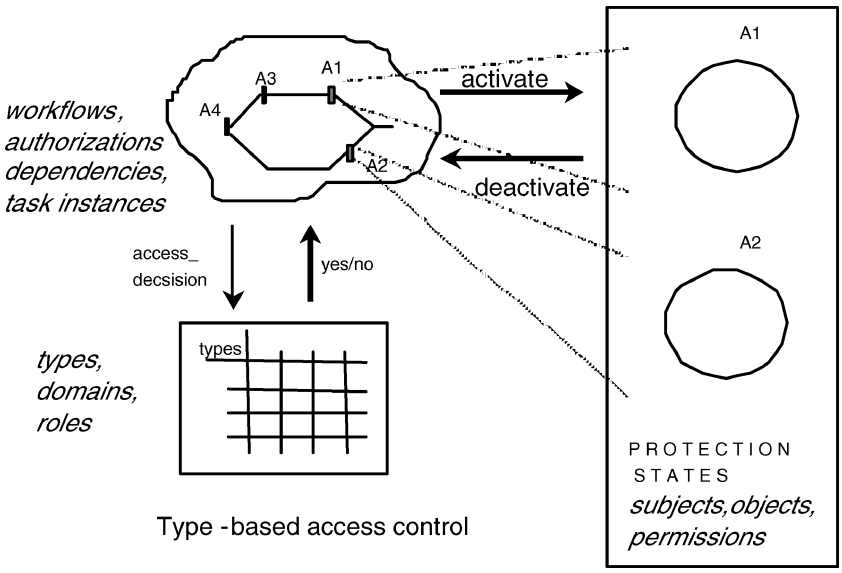

Fig. 5. Task-based access control.

on first-order logic to formally specify policies and conditions that determine role activation or deactivation, where Temporal-RBAC uses triggers for periodic activation and deactivation of roles as well as activation dependencies among roles. RBAC models have been successfully implemented in workflow systems [Bertino et al. 1999; Kang et al. 2001; Ahn et al. 2000; Park et al. 2001] and distributed environments [Ferraiolo and Barkley 1997; Ferraiolo et al. 1999].

3.3. Task-Based Access Control (TBAC)

The TBAC [Thomas and Sandhu 1994, 1997] was introduced to better recognize the broader context in which security requests arise. The TBAC model extends the traditional subject/object-based access control models by including domains that contain task-based contextual information. Access control in this model is granted in steps that are related to the progress of tasks. Each step is associated with a protection state containing a set of permissions. The contents of this set change based on the task. The model, thus, is an active model that allows for dynamic management of permissions as tasks progress to completion. Each step has a disjoint protection state as shown in Figure 5.

TBAC, unlike RBAC, also supports type-based, instance, and usage-based access. In addition, authorizations have a strict runtime usage, validity, and expiration characteristics.

3.3.1. Shortcomings. There are several weaknesses to the TBAC model. We identify some of the important issues as follows.

—TBAC systems recognize the need to incorporate contextual parameters into security considerations; however, it is limited to contexts in relation to activities, tasks, or workflow progress and is implemented mainly by keeping track of usage and validity of permissions. Collaborative systems require a much broader definition of context, and the nature of collaboration cannot always be easily partitioned into tasks with usage counts.

—Permissions are activated and deactivated in a just-in-time fashion. The draw-back of that is that if a central access control module manages

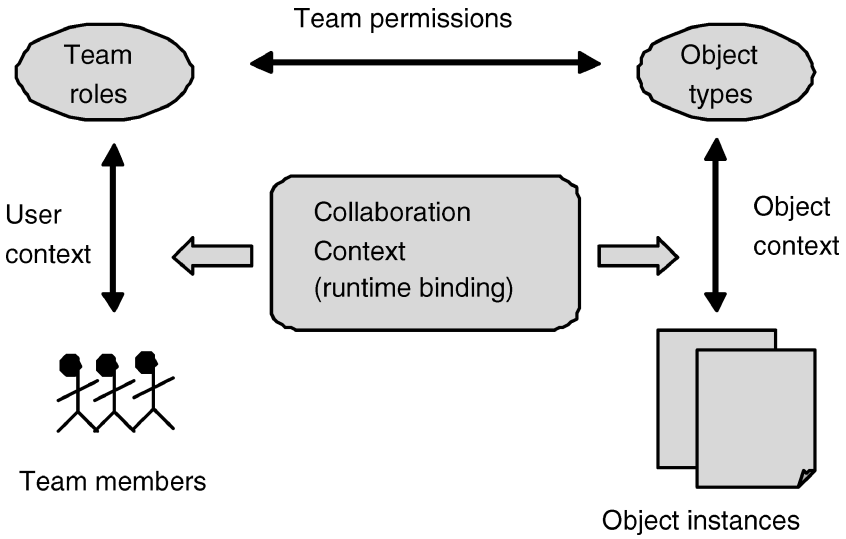

Fig. 6. Team-based access control.

permissions, it could introduce several constraints, such as race conditions, across workflows.

—Specification of complex policies and management and delegation and revocation of authorization privileges in TBAC are very primitive. More fine-grained components need to be defined to support dynamic environments motivated by TBAC.

—TBAC can be used effectively for security modeling and enforcement from an application or enterprise point of view and has its advantages over the system-centric approach in subject-object systems. But for most collaborative environments, TBAC is used within other access control models.

3.3.2. Collaborative Frameworks. Kang et al. [2001], Ahn et al. [2000], and Park et al. [2001] used TBAC and RBAC to define access control mechanisms for interorganizational workflow. Roles are used as an interface between workflows and security infrastructures specific to organizations. The PerDis Groupware Platform [Coulouris et al. 1998] that implements a persistent distributed store also uses an access control model based on roles and tasks.

3.4. Team-Based Access Control (TMAC)

The notion of access rights associated with groups of users rather than individuals is a feature that was recognized early in the investigation of requirements for access control in collaborative environments. In RBAC models, groups are defined on the basis of users belonging to the same role. While the strengths of this approach have been discussed, there is a limitation when one considers instances of roles collaborating on group work. Teams, on the other hand, appear to be a more natural way of grouping users in an enterprise or organization and associating a collaboration context with the activity to be performed.

The TMAC model proposed by Thomas [1997] defines two important aspects of the collaboration context, user context and object context. Figure 6 shows these components in TMAC. User context provides a way of identifying specific users playing a role on a team at any given moment, and object context identifies specific objects required for collaboration purposes.

TMAC offers the advantages of RBAC along with provisions to specify fine-grained control on individual users in certain roles and on individual object instances. As a further extension to this approach, Context-based TMAC (C-TMAC)

[Georgiadis et al. 2001] integrates RBAC and TMAC by incorporating context as an entity in the architecture. C-TMAC seeks to include contextual information other than user and object contexts such as time, place, and so forth.

3.4.1. Shortcomings. TMAC and C-TMAC have very unique features to support contextual information and the dynamic nature of team-based environments. However, there are several weaknesses to these models. Here is a list of identified weaknesses.

—TMAC and C-TMAC both extend RBAC with the notion of a team. However, the models have not yet been fully developed, and it is not clear how to incorporate the team concept into a general RBAC framework.

—TMAC and C-TMAC lack the self-administration of assignment relations between entities. These model also need to reflect the multidimensional definition of rich collaborative contexts such as organizational entities, workflow tasks, groupware's environmental components, and so on. The fine-grained administration of TMAC and C-TMAC entities and relations is necessary to demonstrate applicability and usability of these models.

—Wang [1999] suggests the use of both Team- and Role-Based Access Control for Hypermedia environments. The model's general applicability, however, requires additional testing in more realistic settings. The specification, selection, and application of policies in multiple settings also require further investigation.

3.5. Spatial Access Control

Bullock and Benford [1999] propose a spatial access control for collaborative virtual environments, SPACE [Bullock 1998]. This model takes into account the environment of collaboration and exploits it to hide explicit security mechanisms from the end users. The model consists of two main components: a boundary and an access graph. The boundary divides the collaborative environment into regions and accounts for both traversal as well as awareness within regions. It uses the notion of credentials to allow access within regions. The access graph is used to specify the constraints on movement in the collaboration space as well as the management of the access requirements in the environment.

3.5.1. Shortcomings. This model is concerned only with navigational access requirements in collaborative environments and does not provide for fine-grained control. The SPACE model is not provably secure, unlike other access control models. It is possible for users to apply the SPACE model and create insecure regions. This model lacks the complexity needed for systems where the level of security provided is important. This model cannot be used unless it is possible to represent an application in terms of regions and boundaries.

3.5.2. Collaborative Frameworks. The SPACE model was originally created for 3D Spatial, Collaborative, Virtual Environments, but it has been shown to be applicable in several other areas as well. It was applied to graphical, collaborative, virtual environments like Spline [Bullock 1998]. In Bullock and Benford [1999], there is also mention of using SPACE for 2D CSCW applications that contain spatial structuring.

3.6. Context-Aware Access Control

Covington et al. [2001] have extended RBAC with the notion of environment roles in order to provide for security in context-aware applications. They use roles, called environment roles, to capture environmental state. These roles use role hierarchy, activation, separation, and so on, similar to traditional RBACs, to manage the policies and constraints that depend on the context of collaboration. These roles are activated based on environment conditions at the time of request.

The environment RBAC has been shown to be of use in ubiquitous computing where environment-sensitive information is pervasive. This approach, however, requires further testing within the collaborative systems domain.

3.7. Reflections on the Evolution of Access Control Models

From preliminary work in Access Matrix- and RBAC-based models to recently proposed frameworks, traditional access control models have been extended using a variety of concepts. One of the most significant characteristics of these efforts is an emerging recognition of the importance of utilizing contextual information in authorization decisions. Context is one of the most defining aspects of collaborative environments because it encapsulates not only all types of environmental variables (participants, resources, tasks, etc.), but also the dynamism and unpredictability associated with them. This, as well as the fact that context also embodies parameters such as time, place, presence, copresence, awareness, and so on, is leading designers to incorporate broader notions of context into models for access control. The concept of users in a set of activated roles in a session is well accepted (see Section 3.2 RBAC). Context introduces another level of consideration because a user in different roles could be active in different contexts, and these contexts could change with his/her location, presence of other users in the same location or remote location, the roles in which other users are present and active, and so on. For instance, it is insufficient to state that a user can perform an operation only if s/he is active in a particular role. Rather, it may be the case that the user can perform the operation only if his/her current context includes (or possibly does not include) other users active in specific roles. For example, a professor might be able to read/view a grade sheet within a shared collaboration space as long as no students are simultaneously present in that same space.

4. ASSESSMENT CRITERIA FOR ACCESS CONTROL IN COLLABORATIVE SYSTEMS

We have described several access control models that have been proposed or used for collaborative environments. In this section, we try to evaluate these models against a set of criteria relevant to access control models in collaborative environments. These criteria have been drawn from the characteristics of access control models in collaborative systems mentioned in Section 3. The criteria we used to characterize the access control models are as follows.

—Complexity defines the nature of the access control model. It is considered an important aspect of consideration because an overly complex model can lead to unforeseen problems and implementation can become difficult. There is a tradeoff between functionality and complexity.

—Understandability defines the transparency of the model and its underlying principles. The consequences of manipulation and changes of access rights should be obvious for the proper use of the system.

—Ease of use indicates how simple the system is from the end user's point of view in terms of its usage in a collaborative environment. If it is very cumbersome to use, then there is a chance that users will not favor it. Security systems always bring a degree of complexity into the system, and users need to be reassured of the ease of use of any system. The simpler the model is, the more popular it will be.

—Applicability of an access control model is an indication of its practicality. A good, but solely theoretical, model may provide few benefits. An infrastructure should exist where the model can be deployed.

—Support for collaboration is the most important aspect of consideration for access control models devised for collaborative environments. There are several aspects of a collaborative environment that determine the ultimate usability of

Table 1. Characterization of Access Control Models for Collaborative Systems

Criteria	Matrix	RBAC	TBAC	TMAC	C-TMAC	SAC	Context-AW
Complexity	Low	Medium	Medium	Medium	Medium	Low	High
Understandability	Simple	Simple	Simple	Simple	Simple	Simple	Simple
Ease of Use	Medium	High	Medium	High	High	Low	High
Applicability	Medium	High	Medium	Medium	High	Low	High
Collab. Support:							
Groups of users	Low	Y	Y	Y	Y	Y	Y
Policy Specification	Low	Y	Low	Y	Y	Y	Y
Policy Enforcement	Low	Y	Low	Y	Y	Low	Y
Fine grained control	N	Low	Low	Y	Y	N	Y
Active/passive	Passive	Passive	Active	Active	Active	Active	Active
Contextual info.	N	Low	Medium	Medium	Medium*	Medium	Medium*

any access control model. These factors are discussed individually.

—*Groups of Users*. A collaborative environment in its most basic form implies a common task undertaken by a group of people. The access control model should represent support for changes, manipulation, and specifications made for groups of users in addition to individual users.

—*Policy Specifications*. Access control models are based on the specification and representation of policies that govern a collaborative environment. The model should support ways of specifying policies and an appropriate syntax, pattern, or language that allows extensions or modifications in a simple and transparent manner. This helps to ensure the scalability of the system.

—*Policy Enforcement*. It is important for the access control model to provide means to ensure that the policies or constraints specified are enforced correctly.

—*Fine-Grained Control*. Collaborative environments are characterized by situations where it is not sufficient to have access rules only for groups of users on clusters of objects. Often, a user in an instance of a role might need a specific permission on an instance of an object at a particular point in the collaboration instance. A level of fine-grained control is required for such situations, without introducing compromises or complexities into the system.

—*Active/Passive*. It is desirable for the access control model to be active so as to handle the dynamism of a collaborative system.

—*Contextual Information*. Context is one of the most important characteristics of any collaboration, and it is important to know the degree to which contextual information is utilized by the access control model in order to secure the system.

Table I evaluates the access control models examined in this article against the criteria mentioned. The table makes use of comparative terminology such as Low, Medium, and High, descriptive terminology such as Simple, Active, and Passive, and the standard Yes (Y) and No (N) terminology for characterization against the criteria.

For the contextual information criteria, Medium* is used to identify those models that appear to support the strongest notion of context among those in the Medium category. Use of Low, Medium, and High for criteria such as Complexity describes the degree. Low Complexity indicates that the model is fairly simple in nature.

Low has also been used to describe criteria such as groups of users when it is not convenient to use a simple Yes or No means of description. For example, ACLs provide ways of specifying access rights for groups of users, but it is very primitive and cumbersome from the point of view of supporting groups in a collaborative environment. Low in such a situation indicates primitive support for the concerned characteristic or feature.

Yes and No have been used whenever it is possible to indicate the facilitation or lack of facilitation of the concerned criteria by the access control model. Wherever it is insufficient to simply indicate the presence of support for a feature, and it is also important to indicate the degree to which a feature is supported, Low, High and Medium have been used. For example, ACLs and traditional RBAC do not support consideration of contextual information in decision-making, whereas the other models support varying degrees of contextual information consideration.

5. CONCLUSION

In this article, we provided a comprehensive study of authorization mechanisms for collaborative environments. Key to this study was an examination of both the merits and weaknesses of each approach, as well as the identification of emerging trends for authorization models for collaboration. We began by presenting access control requirements for collaboration as documented from existing research. Next, we examined existing access control models as applied to collaborative environments in light of these requirements, highlighting not only the benefits, but, more importantly, the weaknesses of these models. Following this investigation, we assessed these models based on criteria drawn from this investigation. We believe that our study helps to introduce a new model for access control drawing upon current best practices in the access control community as well as to evaluate such a model in a collaborative systems environment.

REFERENCES

AHN, G.-J. AND SANDHU, R. 2000. Role-based authorization constraints specification. *ACM Trans. Inf. Syst. Secur. 3*, 4 (Nov.).

AHN, G.-J., SANDHU, R., KANG, M., AND PARK, J. 2000. Injecting RBAC to secure a Web-based workflow system. In *Proceedings of 5th ACM Workshop on Role-Based Access Control*. Berlin, Germany.

AHN, G.-J., SHIN, D., AND ZHANG, L. 2004. Role-based privilege management using attribute certificates and delegation. In *International Conference on Trust and Privacy in Digital Business*. Lecture Notes in Computer Science. Springer-Verlag.

AHN, G.-J., ZHANG, L., SHIN, D., AND CHU, B. 2003. Authorization management for role-based collaboration. In *IEEE International Conference on System, Man and Cybernetic*. Washington, DC. 4128–4214.

BERTINO, E., FERRARI, E., AND ATLURI, V. 1999. Specification and enforcement of authorization constraints in workflow management systems. *ACM Trans. Inf. Syst. Secur. 2*, 1 (Feb.).

BERTINO, E., FERRARI, E., AND BONATTI, P. A. 2000. TRBAC: A temporal role-based access control model. In *Proceedings of 5th ACM Workshop on Role-Based Access Control*. Berlin, Germany.

BROTHERS, L., SEMBUGAMOORTHY, V., AND MULLER, M. 1990. Icicle: Groupware for code inspection. In *ACM Conference on Computer-Supported Cooperative Work*. Los Angeles, CA. 169–181.

BULLOCK, A. 1998. SPACE: Spatial access control for collaborative virtual environments. PhD. thesis, University of Notingham.

BULLOCK, A. AND BENFORD, S. 1999. An access control framework for multi-user collaborative environments. In *ACM GROUP*. Phoenix, AZ.

COULOURIS, G., DOLLIMORE, J., AND B., R. 1998. Role and task-based access control in the perdis groupware platform. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*. Fairfax, VA. 115–121.

COVINGTON, M., LONG, W., SRINIVASAN, S., DEY, A., AHAMAD, M., AND ABOWD, G. D. 2001. Securing context-aware applications using environment roles. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA.

DEWAN, P. AND SHEN, H. 1998. Flexible meta-access control for collaborative applications. In *ACM Conference on Computer-Supported Cooperative Work*. Seattle, WA.

EDWARDS, W. K. 1996. Policies and roles in collaborative applications. In *ACM Conference on Computer-Supported Cooperative Work*. Cambridge, MA.

ELLIS, C. A., GIBBS, S. J., AND REIN, G. L. 1989. Design and use of a group editor. In *International Federation for Information Processing Working Group 2.7, Working Conference on Engineering for Human-Computer Interaction*. 13–28.

FERRAIOLO, D. AND BARKLEY, J. 1997. Specifying and managing role-based access control within a corporate intranet. In *Proceedings of 2nd ACM Workshop on Role-Based Access Control*. Fairfax, VA. 77–82.

FERRAIOLO, D. F., BARKLEY, J. F., AND KUHN, D. R. 1999. A role based access control model and reference implementation within a corporate intranet. *ACM Trans. Inf. Syst. Secur. 2*, 1 (Feb.).

GEORGIADIS, C. K., MAVRIDIS, I., PANGALOS, G., AND THOMAS, R. 2001. Flexible team-based access

control using contexts. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA.

GRIEF, I. AND SARIN, S. 1987. Data sharing in group work. *ACM Trans. Inf. Syst. 5*, 2 (April), 187–211.

JAEGER, T. 1999. On the increasing importance of constraints. In *Proceedings of 4th ACM Workshop on Role-Based Access Control*. Fairfax, VA. 33–42.

JAEGER, T. AND PRAKASH, A. 1996. Requirements of role-based access control for collaborative systems. In *ACM Role-based Access Control Workshop*. Gaithersburg, MD. 53–64.

KANG, M. H., PARK, J. S., AND FROSCHER, J. N. 2001. Access control mechanisms for inter-organizational workflow. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA.

KAPLAN, S., TOLONE, W., D.P., B., AND BIGNOLI, C. 1992. Flexible active support for collaborative work with conversation builder. In *ACM Conference on Computer-Supported Cooperative Work*. Toronto, Ontario, Canada. 378–385.

LAMPSON, B. 1971. Protection. In *5th Princeton Symposium on Information Science and Systems*. 437–443. Reprinted in *ACM Operat. Syst. Rev. 8*,1, 18–24, 1974.

NEUWIRTH, C. M., KAUFER, D. S., CHANDHOK, R., AND MORRIS, J. H. 1990. Issues in the design of computer support for co-authoring and commenting. In *ACM Conference on Computer-Supported Cooperative Work*. Los Angeles, CA. 183–195.

PARK, J., SANDHU, R., AND AHN, G.-J. 2001. Role-based access control on the web. *ACM Trans. Inf. Syst. Secur. 4*, 1 (Feb.).

SANDHU, R. AND SAMARATI, P. 1994. Access control: Principles and practice. *IEEE Communications 32*, 9, 40–48.

SANDHU, R. S., COYNE, E. J., FEINSTEIN, H. L., AND YOUMAN, C. E. 1996. Role-based access control models. *IEEE Computer 29*, 2 (Feb.), 38–47.

SHEN, H. AND DEWAN, P. 1992. Access control for collaborative environments. In *ACM Conference on Computer-Supported Cooperative Work*.

SHIN, D., AHN, G.-J., AND CHO, S. 2002. Role-based EAM using x.509 attribute certificate. In *Proceedings of 16th Annual International Federation for Information Processing Working Group 11.3, Working Conference on Data and Application Security*. Cambridge, UK.

SIKKEL, K. 1997. A group-based authorization model for cooperative systems. In *ACM Conference on Computer-Supported Cooperative Work*. 345–360.

SOHLENKAMP, M. AND CHWELOS, G. 1994. Integrating communication, cooperation, and awareness: The diva virtual office environment. In *ACM Conference on Computer Supported Cooperative Work*. Chapel Hill, NC. 331–343.

THOMAS, R. 1997. Team-based access control (TMAC). In *Proceedings of 2nd ACM Workshop on Role-Based Access Control*. Fairfax, VA. 13–19.

THOMAS, R. AND SANDHU, R. 1997. Task-based authorization controls (TBAC): Models for active and enterprise-oriented authorization management. In *Database Security XI: Status and Prospects*, T. Y. Lin and X. Qian, Eds. North-Holland.

THOMAS, R. AND SANDHU, R. S. 1994. Conceptual foundations for a model of task-based authorizations. In *Proceedings of 7th IEEE Computer Security Foundations Workshop*. Franconia, NH. 66–79.

WANG, W. 1999. Team-and-role-based organizational context and access control for cooperative hypermedia environments. In *ACM Hypertext*.

YAO, W., MOODY, K., AND BACON, J. 2001. A model of oasis role-based access control and its support for active security. In *ACM Symposium on Access Control Model and Technology*. ACM. Chantilly, VA.

ZHANG, L., AHN, G.-J., AND CHU, B. 2001. A rule-based framework for role-based delegation. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA. 153–162.

ZHANG, L., AHN, G.-J., AND CHU, B. 2003. A rule-based framework for role-based delegation and revocation. *ACM Trans. Inf. Syst. Secur. 6*, 3 (Aug.).

Received September 2002; revised November 2003; accepted January 2005