AN ADMINISTRATIVE MODEL FOR ROLE GRAPHS

He Wang and Sylvia L. Osborn *
Department of Computer Science, University of Western Ontario, Canada

Abstract For the role graph model, we have always assumed that the administration of the role graph and group graph is carried out by a single (centralized) administrator. This is not realistic in practise. In this paper, we present a decentralized administrative model for the role and group graphs. The model uses administrative domains, and administrative roles which are part of the role graph. A discussion of why domains cannot overlap is given, as well as a discussion of how users and privileges might be in more than one domain. Details of administrative privileges are given. We also present a detailed discussion comparing the new model to other administrative models for role-based access control.

Keywords: role-based access control, administration of access control

1. Introduction

The role graph model has been described in previous papers [6, 7], and is one version of role-based access control (RBAC). The model is described on three planes: the user plane on which users and groups can be modeled [9], the privileges plane on which implications among privileges can be modeled [3], and the role plane on which the role graph is built. The role graph represents role-role relationships by an acyclic, directed graph called the role graph. A tool has been developed which allows users to interact with algorithms for a role graph and a group graph, and to assign users/groups to roles [10].

Our previous papers have always assumed centralized administration of the role graph. This is not realistic in most applications of role-based access control. The motivation for using RBAC is that it helps with management of access control in an enterprise with many (i.e. thousands

*This research was supported by the Natural Sciences and Engineering Research Council of Canada.

of) users, roles and privileges. Such an environment is not manageable by one person. The purpose of this paper is to introduce decentralized administration for the role graph model.

Crampton and Loizou [1] and Sandhu et al. [12, 13, 8] have presented administrative models for RBAC. ARBAC97, presented in [13], discusses users, roles and privileges, how they interact, and how they can be managed. ARBAC02, in [8], modifies the user and permission aspects of their previous administrative model. Crampton and Loizou develop a concept called an administrative scope, which changes dynamically as the role hierarchy changes. Recently, others have also introduced administrative aspects to RBAC models [4, 2, 15]. We will give a more detailed discussion of these other models after introducing our own model.

The paper is organized as follows: Section 2 reviews the basic concepts of the role graph model. Section 3 introduces the administrative model. Section 4 gives descriptions of administrative operations on regular roles. Section 5 discusses operations on administrative domains and other administrative privileges, and includes a simple example showing how we get started. Section 6 contains a detailed comparison with other models. Conclusions are found in Section 7.

2. The Role Graph Model

In the role graph model [7], roles consist of a role name and a set of privileges, each of which is an (object, operation) pair. Roles are arranged in a role graph, with two distinguished roles: MaxRole and MinRole. MaxRole represents all the privileges in the role graph and need not be assigned to any user or group. MinRole represents the least privileges assigned to anyone in the system. We distinguish between *direct privileges* which are those directly assigned to a role, and *effective privileges* which consist of the direct privileges and those inherited from junior roles. The effective privilege set represents all privileges available to any user assigned to a role. Role r_i *is-junior* to role r_j if $\text{effective}(r_i) \subset \text{effective}(r_j)$. Role graphs have the following properties:

- there is a single MaxRole,
- there is a single MinRole,
- the graphs are acyclic,
- there is a path from MinRole to every role r_i,
- there is a path from every role r_i to MaxRole,
- for any two roles r_i and r_j, if $\text{effective}(r_i) \subset \text{effective}(r_j)$, then there must be a path from r_i to r_j,
- by convention we draw the graphs with MaxRole at the top, MinRole

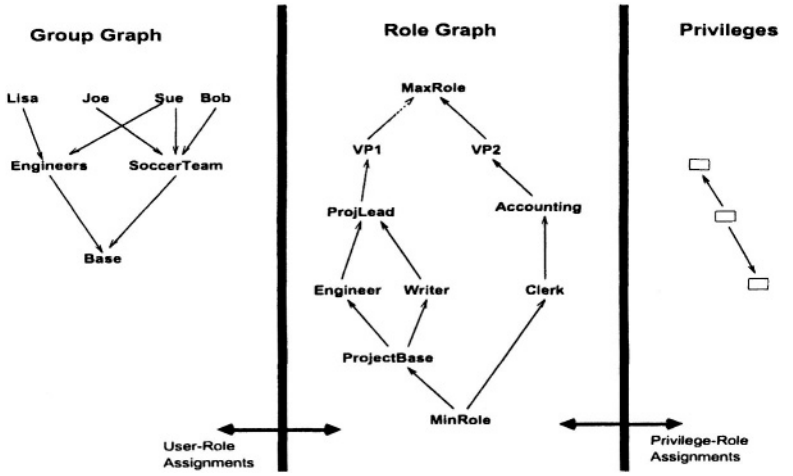

Figure 1. The Three-Plane Model

at the bottom, and junior roles lower on the page than their seniors.
- we remove transitive edges from the display to make the graph less cluttered.

The Group Graph model allows one to create sets of users, say to represent committees or people assigned to a project, who may not have the same job title. To simplify the model, each individual user is regarded as a group of cardinality 1. The edges in the group graph are determined by the subset relationship between two groups. By convention we draw the group graph with the Base group which contains all users at the bottom, and the groups representing individual user groups at the top.

Conceptually, we view the model on three planes, as shown in Figure 1. The structure of the group graph is developed on the left plane, the role graph is developed in the middle plane, and the privileges are represented in the right hand plane. Privileges are assigned to roles by operations on the role graph. We assume that if there are implications among privileges, all implied privileges are included when a privilege is assigned to a role [3]. Users are assigned to roles by considering the interaction between the group graph and the role graph. Other operations are available on role graphs to insert/delete roles, insert/delete edges and add/remove privileges from a role. Any of these role graph operations may alter the "shape" of the role graph when the role graph properties above are restored. As well, all of the operations abort and leave the role graph unchanged if a cycle would be created.

3. The Administrative Model

The operations already available in the role graph model were motivated by considering what might be useful to an administrator when trying to design the access control for a complex system. The structure of the administrative model presented here has been motivated by considering a large corporate environment and the hierarchical management style that is likely to exist in such an environment. We would like the administration to be captured by administrative roles, which are a (special) part of the role graph. Special privileges will be developed to allow the administrative roles to carry out their administrative operations. The administrative model needs to be aware of the algorithms already available for manipulating role graphs, and possibly put constraints on the extent to which a given security administrator may make changes to the role graph. At the same time, the role graph properties must also be maintained whenever an administrative role alters the role graph.

People in the business world usually draw a hierarchical organization structure called an organization chart or reports-to graph, which can also be called a supervision hierarchy [5]. The root node of this tree is the president or CEO; the lowest nodes are positions in departments or branches. This hierarchical structure is not the same as an RBAC role inheritance structure or role graph, because the higher level positions may have fewer privileges than the lower level positions. For example, the president of the company may only have privileges to carry out some very high level operations, and may not be permitted to look at all the details of all the data under control of the RBAC system.

In a typical company, access control is administered in a decentralized way by a group of administrators, security officers and other employees [11]. At the top, there is a CEO or CFO in charge of all security issues of the company. Under the CEO or CFO is an SSO (System Security Officer) whose responsibility is the creation and enforcement of the company security policy. Depending on the complexity of the company, there might be department security officers (DSOs) or branch security officers (BSOs). They represent the local security administrators for possibly decentralized parts of the company. Our administrative model must be able to describe such a decentralized management of access control.

Our administrative model is built on the idea of an administrative domain. Administrative domains (or admin domains) have some relationship to Crampton and Loizou's administrative scope and Sandhu's role range, but they have some important differences. The basis of our model is still the role graph, and the administrative roles will also be part of the role graph. The main idea is to divide the role graph into

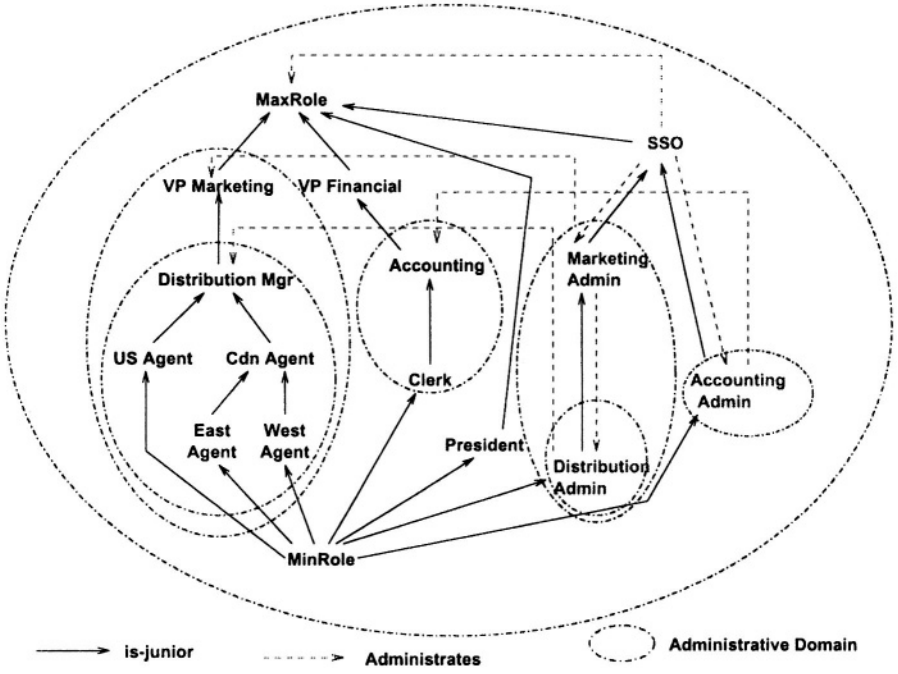

Figure 2. Role Graph with Administrative Domains

administrative domains in which operations can be carried out to alter the role graph, or users and privileges can be assigned to roles. We also need a way to relate administrators to the portion of the role graph that they are going to administrate.

An *Administrative Domain* is a set of roles which contains a top-most role d called the *domain identifier* or domain ID, and all roles in the role graph s such that s is-junior d, except for MinRole. In the role graph in Figure 2, several administrative domains are shown. As well, there are administrative roles: SSO, Marketing Admin, Distribution Admin and Accounting Admin. Edges showing the relationship between administrative roles and administrative domains are also shown in the Figure. This is a many-to-many relationship; for example the SSO administrates several domains in the example. The Distribution domain in the example is administrated by several administrative roles: Distribution Admin, and indirectly by Marketing Admin and SSO.

The model also includes a single highest admin domain called the *default domain,* which includes MaxRole and MinRole. The administrative role assigned to administrate this domain should be the highest administrative role in the company. This is the only administrator who can

make changes to MinRole. In previous presentations of the Role Graph Model, this highest administrative role was assumed but not shown, and this was the only admin domain present. Thus this administrative model is consistent with the original Role Graph Model.

Consider the intersection I of two administrative domains, $I = D_1 \cap D_2$. If I is non-empty, then if $I = D_1$ or $I = D_2$, we say the domains are *nested.* If I is non-empty and not equal to D_1 or D_2, we say we have an *overlap. We do not allow administrative domains to overlap.* The reason for this is the following. Suppose, looking at Figure 2, there is a role X which comes between MinRole and West Agent, and between MinRole and Clerk. By the definition of administrative domains, this role would be in the accounting, marketing and distribution domains. The administrators of any of these domains could add or delete privileges to/from X. Thus these administrators could make changes to the role graph which would affect domains of which they are not the administrator. Thus, overlapping domains are not allowed. If there is some basic set of privileges which all domains need, they can be added to MinRole. The case where only some domains need a common privilege is discussed below.

Just as we do not want roles that exist in more than one domain, we also do not want edges that go from one domain to another. This can arise if privileges are allowed to be in more than one domain, which we *would* like to allow. There is always some data which may need to be shared between domains. For example, if each admin domain represents a branch of a large bank, the information about the bank's savings account customers should be available in every branch. Having such shared data means that there is probably some basic privilege like (customer data, read) which represents the sharing of the data. Creating a role with only this one privilege in domain D_1, would cause an edge to any role in domain D_2 which contains this same privilege. Having an edge from a role r_1 in domain D_1 to a role r_2 in domain D_2 means that if another privilege is added to role r_1, the role graph structure and algorithms will cause this privilege to be inherited by r_2, which again means that the administrator of D_1 is making changes that affect D_2.

There are several ways to deal with these shared privileges. One way is to put them all in MinRole. This may not always be feasible since they would also be inherited by every other domain. A second way, currently being implemented in a prototype [14], is to extend the model for privileges so that each privilege also has a domain ID. Privileges must have the same domain ID as the domain of any role they become part of. We can still equate the privileges by looking at the object name and access mode.

The third way to deal with shared privileges is to not have a domain ID attached to each privilege, but to check after each operation on the role graph if any edges have been created which cross domain boundaries, and if this is so, abort the operation (just as the current role graph algorithms abort any operation if a cycle would be created). If the security administrators are careful to always put these shared privileges into roles with some privilege unique to the domain, then edges between domains should never result.

Administrative domains also include users and groups from the group graph (not shown in Figure 2). We want it to be possible for a user to work in different domains (an employee might work at different branches of a bank on different days of the week), so that overlapping of the users in different domains is allowed. In the current prototype, each user has one or more domain IDs associated with it.

4. Administrative Operations on Regular Roles

Administrative roles, like other roles, have privileges. Since they are just roles, these privileges could be anything. However, in keeping with the desire to keep the model intuitive to people running real companies, these privileges should be restricted to being administrative privileges. Algorithms for these operations have been presented and implemented for a role graph with centralized administration [7, 10]. They have to be modified in order to enforce the concept of administrative domains. We have to add constraints to the role graph operations so that the operations permitted do not alter parts of the role graph outside of an administrative role's administrative domain. Note that modifying the group memberships or group-role assignments does not alter the role graph, but modifying privilege-role assignments can alter the role graph quite significantly.

Here is a list of the operations needed and how they must be modified to maintain the desired properties of administrative domains.

Add a privilege to a role: If an arbitrary privilege is added to a role, it will, by the inheritance caused by the is-junior relationships, be inherited all the way up to MaxRole. This is an example of where a local operation can affect parts of the graph outside of an administrative domain. We solve this problem by having an administrator of an enclosing domain (there is always one such domain, the default domain) add/remove privileges from the domain identifier, i.e. the top role in each domain, as direct privileges if necessary. Then the domain administrator is constrained to choosing from the domain ID's privileges when adding a privilege to a lower role in the domain. Such a privilege ad-

dition does not alter the effective privileges of the domain ID, and thus does not affect roles outside of the administrative domain. By adding a privilege to a role like West Agent (a privilege already available in Distribution Mgr) in Figure 2, it becomes a direct privilege of West Agent, and an effective privilege of role Cdn Agent. The domain administrator is prevented from adding privileges to the domain ID (since this must be done by a more senior administrator).

If the third technique for handling overlapping privileges is being used, then this operation, and all the other operations in this list, must be rejected if edges between domains result.

Delete a privilege from a role: By the same arguments as just given, privileges may be deleted from an arbitrary role in a domain, but they will remain in the direct privilege set of the domain ID. Only an administrator of an enclosing domain may delete privileges from the domain ID.

Add a role: There are two role addition algorithms given in [7]. The first one gives a role name, set of direct privileges, set of proposed immediate juniors, and proposed immediate senior roles. The constraints that need to be imposed now are that the privileges must all be in the effective privilege set of the domain ID, and the proposed juniors and seniors must all be roles in the domain or MinRole.

The second role addition algorithm gives the new role name, and proposed effective privileges. The role graph algorithm figures out its juniors and seniors. The constraint needed now is that the proposed effective privileges must all be in the effective privilege set of the domain ID. In the absence of privileges shared among domains, any juniors or seniors that the algorithm detects will all be within the specified domain.

Delete a role: The current algorithm gives the user the choice of keeping the privileges in the deleted role's seniors, or removing the privileges from the graph. In the administrative version, even if the privileges are removed from seniors, they must remain in the domain ID.

Insert an edge: Inserting edges can make privileges be inherited by roles where they did not previously exist. To avoid making changes outside of the administrative domain, inserted edges must involve two roles from the administrative domain being altered. This can include the domain ID.

Delete an edge: For similar reasons, deleted edges must have both roles from the administrative domain being altered.

Assign a user or group to a role: Both the role and the user or group must be in the same domain.

Remove a user or group from a role: Both the role and the user or group must be in the same domain.

These operations together with a domain ID become privileges that can be assigned to the appropriate administrative roles.

5. Operations on Administrative Domains

In addition to the above privileges which can be assigned to administrative roles, we need operations to create, destroy etc. whole domains. Important to these operations is a test of whether or not two domains overlap. We now discuss the details of these operations.

Create a new domain given r, its domain ID: This operation may be executed by an administrative role of an administrative domain D which contains r such that r is not the domain ID of D. To create the domain, we need to perform a depth-first traversal to all juniors of r, except for MinRole, and then check that the resulting domain does not overlap any existing domain.

Delete a domain given r, its domain ID: This operation may be executed by an administrative role of an administrative domain which contains r such that r is not the domain ID. All record of this domain must be removed, including any administrative privileges on it held by administrative roles.

Assign an administrative role to an administrative domain: This operation deals with how the "Administrates" edges appear in the graph. Administrative roles can form an (administrative) domain by themselves, or as in the figure, can be the domain ID of another administrative domain (by the definition of administrative domain, Marketing Admin and Distribution Admin in Figure 2 form an administrative domain). In order to assign, for example, the Distribution Admin administrative role to the administrative domain whose ID is Distribution Mgr, the role responsible must be in an administrative position with respect to both the admin role and the domain. So, in this example, this could be performed by Marketing Admin or SSO. In general, then, this operation can be performed by an administrative role which administrates both the domain and the admin role in the operation. The topmost admin role (SSO in the example) has to be able to assign itself as administrator of junior admin roles. This is also consistent since SSO administrates the default domain, which includes itself.

Assign an administrative privilege to an administrative role: This operation also can be performed by an admin role which administrates both the domain and the admin role involved. It may be that not all the possible administrative privileges listed in the previous section should be assigned to every administrative role. Thus, for example, we could build admin roles which can only assign users/groups to roles but

cannot alter the role graph (to represent the Human Resources Department, for example).

Remove an administrative privilege from an administrative role: Constraints for this operation are as above. Every admin domain has the default admin domain's administrator as an administrator.

5.1. Other Administrative Privileges

In addition to the operations above on domains, there are some other admin privileges required. Initially, the system starts with the graph in Figure 3a. The initial admin role, SSO, needs the ability to create other admin roles, which it has by virtue of being the administrator of the default domain. In general, admin roles can also have the ability to create other admin roles as their own children. This privilege would not be granted to all admin roles. All of the admin privileges are granted at the discretion of an admin role, so the SSO could create a Human Resources (HR) admin role, which could create users, and assign them to domains, whereupon the administrators of those domains can assign them to roles in the domains. The HR admin role could create other admin roles as its children, and give them the privilege to create users, or other admin roles which are their children. Some of these administrative privileges are, then:

Create a Child Administrative Role: An admin role with this privilege can create sub-admin roles as its own children. The parent role has administrative rights over the newly created child role.

Assign the Create Child Privilege to an Administrative Role: This is also covered by the assign an admin privilege to an admin role operation above.

Create a User with Domain ID: The admin role must be an administrator of the Domain given.

Create a Group in the Group Graph with Domain ID: The admin role must be an administrator of the domain. All users in the group must be in the domain.

Add a domain ID to a user: The user may have additional domain IDs assigned. The admin role performing this must have an admin relationship with the user and the new domain.

Remove a domain ID from a user: The admin role performing this must have an admin relationship with the user and the domain.

5.2. Building up the Example

The system would start with the graph in Figure 3a. The SSO can add regular roles and admin roles, giving the second graph in Figure 3b.

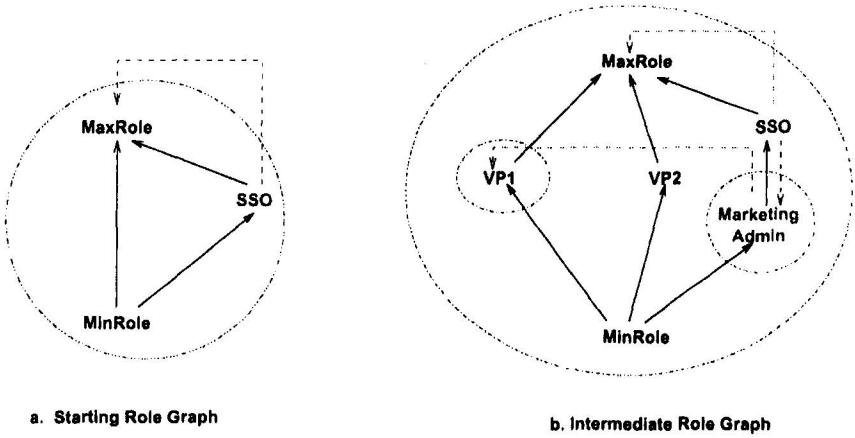

Figure 3. Building the Role Graph

From this graph, either Marketing Admin or SSO can create Distribution Mgr with the appropriate privileges, and the Distribution Admin admin role. SSO can also create the Accounting role and the Accounting Admin role. From this the graph in Figure 2 can be created.

6. Comparison with Other Models

We will begin by comparing our model with that of Sandhu [13, 8]. There are of course many similarities between these two RBAC models. Our role graph corresponds to the role hierarchy in RBAC96 and all subsequent versions of their model, our groups correspond to the groups in RRA97 which are roles with users but no permissions assigned, and our privileges correspond to abilities in RRA97. In fact, abilities model the implications among privileges [3], in that they form a bundle of privileges which should be assigned together to roles. Our algorithm in for following privilege implications gives the detail of how to put together their abilities [3]. Our group graphs provide more detail about how the groups are related. Nowhere, in the Sandhu papers, is there a discussion of anything equivalent to restoring the role graph properties. There is no explicit mention of adding a permission to a role, although deleting a permission is mentioned. Both adding and deleting permissions can cause the effective privileges of two roles higher in the graph to become identical, in which case a cycle in the graph would be created. No mention of how to deal with this is made in the Sandhu papers.

When it comes to the administrative model, URA97 [12, 13], in order to control user-role assignment, uses the idea of prerequisite roles. Clearly some way of constraining which users can be assigned is needed,

but we feel it is not necessarily best modeled by being already in a role. This is replaced in ARBAC02 by the idea of a user pool, which is populated by the Human Resources department. Our idea of requiring that the user be labeled by the appropriate domain ID is meant to model the real-world situation of the user reporting to some manager. It can also correspond to the formation of user groups in our group graph model. The domain ID labeling of users is very similar to ARBAC02's user pool.

For permission-role assignments, PRA97 [12, 13] uses prerequisite roles, which are replaced by permission pools in ARBAC02 [8]. Our idea is to have the usable privileges be those defined as effective privileges in the domain ID role. We give very good motivation for this in discussing how it works together with the algorithms in the role graph model. The effective privileges of the domain ID can be regarded as a permission pool.

In general, in ARBAC97 [12, 13], administrative roles are given control over a role range, which informally has a single top and a single bottom node. We see no good motivation for having a single bottom to a role range. In a business environment, managers are given control over segments of the company. These segments may have subsegments, but the senior manager can usually have control over the finest details, which might also be in the control of a junior manager. Thus, our domains, have a bottom which stops just above MinRole; we feel this more realistically models what happens in real companies. Our discussion of the administrative operations shows no problems with this "open bottom" model when taking into account the role graph algorithms and properties.

Crampton and Loizou's model has a notion of an administrative scope for administrative operations [1]. Again, informally, this scope has a single bottom. The scope also changes dynamically to conform to a mathematical definition. We feel that our administrative domains, which do not necessarily have a single bottom, describe more realistically what happens in real companies. Moreover, the extent of the administrative domain should be the choice of the designer of the administrative model, and not a mathematical accident.

The Enterprise RBAC Model [4] uses the term scopes to refer to collections of objects and administrative permissions. Scopes are arranged in a DAG. Administrative permissions apply to a scope and all its subscopes, or can be restricted to the one scope node, or can be excluded for a scope. To some extent, this corresponds to not necessarily assigning all possible admin privileges to every admin role in our model. There is no discussion of the problems of overlapping scopes.

In [15], Wedde and Lischka describe a model for cooperative, distributed administration of an RBAC system. Their model is built on the idea of authorization spheres, which in turn consist units composed of subjects and objects. Units are arranged in a hierarchy which is different from the role hierarchy. Authorization spheres do not overlap, although no motivation is given for this.

In the role control center model of Ferraiolo et al., [2], administration is defined with respect to views. Views in turn are defined in terms of all seniors of a set of principles. In our terminology, views could have a single bottom and open top. Views are allowed to overlap. This this model is quite different from ours.

7. Conclusions

We have introduced an administrative model for the role graph model, which consists of administrative domains and administrative roles. There are several innovations in this new model.

The administrative model clarifies the relationships between administrative roles and regular roles. The regular roles are contained in administrative domain that is managed by an administrative role. An administrative role can manage its child administrative roles. We also show the administrative relations between the admin roles.

The administrative model described here supports decentralized administration. Our previous model assumed the administration is centralized, but large enterprises require decentralized control over access to data.

The model also fine tunes the role graph by adding new definitions, properties, rules and algorithms. In addition to the admin domain definition, we introduced the definition of administrative roles. Domains cannot overlap in the role plane but can overlap in the user/group plane and privilege plane. The problem of domain overlap is examined in detail; solutions for solving the problem caused by overlapping privileges are discussed. Required modifications of old algorithms are described. New algorithms that manage domain and administrative roles are also discussed.

We also compared our new model with other models in the literature. Our definition of administrative domain uses a “single top, open bottom” approach. We believe this approach is successful because it closely matches business management models. Our administrative model was designed based on a detailed analysis of business management. It should be easily understood by administrators and managers in the business world.

References

[1] Jason Crampton and George Loizou. Administrative scope and role hierarchy operations. In *Proc. 7th ACM SACMAT,* pages 145–154, 2002.

[2] D.F. Ferraiolo, G-J. Ahn, R. Chandramouli, and S.I. Gavrila. The role control center: Features and case studies. In *Proc. 8th ACM SACMAT,* pages 12–20, 2003.

[3] Cecilia M. Ionita and Sylvia L. Osborn. Privilege administration for the role graph model. In *Proc. IFIP WG11.3 Working Conference on Database Security,* July 2002.

[4] A. Kern, A. Schaad, and J. Moffett. An administration concept for the enterprise role-based access control model. In *Proc. 8th ACM SACMAT,* pages 3–11, 2003.

[5] Jonathan D. Moffett and Emil C. Lupu. The uses of role hierarchies in access control. In *Proceedings of the Fourth ACM Workshop on Role-based Access Control,* pages 153–160, 1999.

[6] M. Nyanchama and S. L. Osborn. Access rights administration in role-based security systems. In J. Biskup, M. Morgenstern, and C. E. Landwehr, editors, *Database Security, VIII, Status and Prospects WG11.3 Working Conference on Database Security,* pages 37–56. North-Holland, 1994.

[7] M. Nyanchama and S. L. Osborn. The role graph model and conflict of interest. *ACM TISSEC,* 2(l):3–33, 1999.

[8] Sejong Oh and Ravi Sandhu. A model of role administration using organization structure. In *Proc. 7th ACM SACMAT,* pages 155–162, 2002.

[9] S. Osborn and Y. Guo. Modeling users in role-based access control. In *Fifth ACM Workshop on Role-Based Access Control,* pages 31–38, Berlin, Germany, July 2000.

[10] Sylvia L. Osborn, Yan Han, and Jun Liu. A methodology for managing roles in legacy systems. In *Proc. 8th ACM SACMAT,* 2003.

[11] Thomas R. Peltier. *Information Security Policies, Procedures, and Standards: Guidelines for Effective Information Security Management.* Auerbach Publications, 2001.

[12] R. Sandhu and V. Bhamidipati. Role-based administration of user-role assignment: The URA97 model and its oracle implementation. In *Proceedings of IFIP WG 11.3 Eleventh Annual Working Conference on Database Security,* pages 239–254, Aug. 1997.

[13] R. Sandhu, V. Bhamidipati, and Q Munawer. The ARBAC97 model for role-based administration of roles. *ACM Trans. on Information and Systems Security,* 2(1):105–135, Feb. 1999.

[14] He Wang. Role graph administration in an enterprise environment. Master's thesis, Dept. of Computer Science, The University of Western Ontario, 2003.

[15] H.F. Wedde and M. Lischka. Cooperative role-based administration. In *Proc. 8th ACM SACMAT,* pages 21–32, 2003.