The Uses of Role Hierarchies in Access Control

Jonathan D. Moffett
Department of Computer Science, University of York
Heslington, York YO10 5DD, UK
jdm@cs.york.ac.uk

Emil C. Lupu
Department of Computing, Imperial College
180 Queen's Gate, London SW7 2BZ, UK
e.c.lupu@doc.ic.ac.uk

Abstract

The value of role-based access control (RBAC) is now well recognised. One aspect of it is the ability to make access decisions based upon the position of a role in a hierarchy. It is now recognised that there are some problems associated with this, because of the risk that these decisions may conflict with the control principles that are applied within an organisation. The aim of this paper is to identify the possible uses of role hierarchies in simplifying access rules, while remaining within the constraints of organisational control principles. We use the concept of authority state, i.e., the set of fixed and variable policies and rules in the system which influence the Reference Monitor's access decisions. We then consider the uses of role hierarchies in two separate contexts: first, within a static view of the authority state, where role hierarchies may be used by an access control decision facility; and second, as constraints upon permissible changes to the authority state. We conclude that role hierarchies have some possible uses within the static view, but that they are more important as a means of constraining the permissible changes to the authority state. We make proposals for further research on the place of role hierarchies in controlling change.

1 Introduction

The value of roles for access control has been known for some time, but their use was given a new impetus by the work presented in [1] which proposed a framework of reference models for Role-Based Access Control. In that paper the framework was extended to include role hierarchies. The model allows the occupants of superior roles to inherit all the positive access rights of their inferiors, and conversely ensures that the occupants of inferior positions inherit any prohibitions that apply to their superiors. However, the authors of that paper observe that in some situations inheritance of access rights down the role hierarchy may be undesirable.

Furthermore, there may be multiple role-role relationships. Thus, there may be several role hierarchies, each of which may lead to the inheritance of access rights determined by these relationships, between which it is useful to distinguish. In [2] we used a generalisation (*isa*) hierarchy based on professional competencies for the analysis of role hierarchies. On the other hand [3] used a generalisation hierarchy based on an organisation's functional hierarchy. Both these role hierarchies are valid and useful, but it is apparent that the choice of which (or both) to use for the inheritance of access rights will affect the state of rights in the organisation.

In [4] we took this discussion further:

- We outlined the control principles which are applied in many large organisations and their impact on inherited access rights, and came to the conclusion that the interaction of control principles and role hierarchies could have undesirable consequences for access control;
- We pointed out that the generalisation hierarchy is not the only one that could be validly used for a role hierarchy: we gave examples of an aggregation hierarchy based on the subsetting of a role's activities; and a hierarchy which is based on the supervision relationship.

That paper was limited mainly to pointing out the problems; in this one we make an attempt to provide some

Proceedings of the 4th ACM Role Based Access Control (RBAC) Workshop, Fairfax, VA, USA, Oct. 28-29, 1999.

positive guidelines about the appropriate use of inheritance and hierarchies in an access control system.

We approach this as follows: section 2 outlines the different types of role hierarchies; section 3 summarises organisational control principles. Section 4 describes the concept of authority state and sections 5 and 6 discuss the possible uses of role hierarchies in both a static and dynamic view of the authority state. Finally, section 7 reaches some conclusions and outlines directions for further research.

2 Organisational Role Hierarchies

The concept of role is well-established in the literature of sociology. For reasons which become apparent below, we find it useful to distinguish between a position and a role. A position is simply a named placeholder with no semantics attached to it. We regard a position as a group with a single occupant. A role is defined [5] as the set of rights and duties which are assigned to a person who occupies that role. We summarise here the discussion in [4] of the kinds of role hierarchy that might usefully exist in an organisation. We identified three role hierarchies:

- The *isa* role hierarchy, based on generalisation;
- The Activity role hierarchy, based on aggregation;
- The Supervision role hierarchy, based on the organisational hierarchy of positions.

2.1 Generalisation: the "isa" hierarchy

Sandhu's [1] role hierarchy examples use generalisation, also known as the "isa" relationship, based on competencies: e.g., PrimaryCarePhysician *isa* Physician *isa* HealthCareProvider. Each of these roles is more general than the previous one, and they constitute a partial order. See figure 1, extended from [1] in which we show the relationship both as a role hierarchy and as a generalisation.

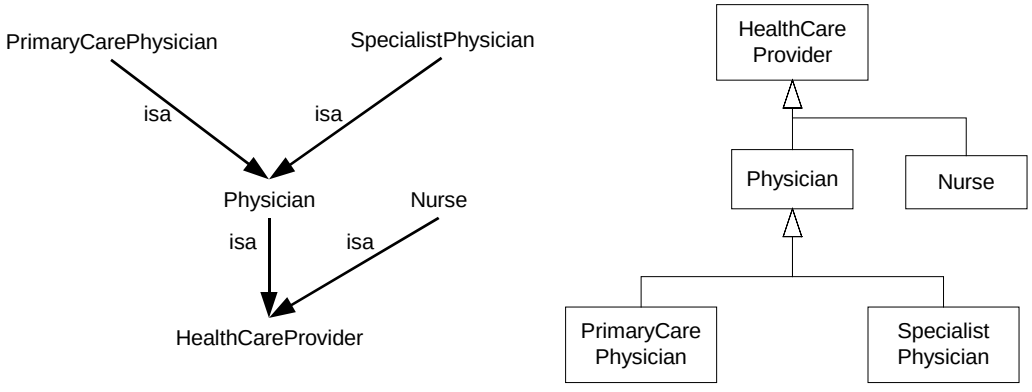
Some of the roles in the *isa* hierarchy may be virtual, i.e., no user occupies them; they are only defined to capture qualities which are shared by real roles further up the *isa* hierarchy. In figure 1 Physician and HealthCareProvider are virtual roles: Physician captures the commonality between PrimaryCarePhysician and SpecialistPhysician; while HealthCareProvider captures the commonality between Physician and Nurse.

Awischus [3] also uses a generalisation hierarchy, but the hierarchy is based on role trees, based on an organisation's functional hierarchy. These role relationships are not necessarily associated with a competency hierarchy.

2.2 Aggregation: the Activity Hierarchy

Aggregation is also known as the "part of" relationship; complex objects are composed of, or aggregated from, parts. A similar concept applies to the activities of an organisation as illustrated in figure 2: the Financial Control activity is composed of Financial Forecasting and Financial Accounting, etc, etc, down to the Accounts Payable and Accounts Receivable activities. The activity hierarchy is partially ordered by subsets of activities.

It is possible to define a role hierarchy based on activities. We can define *ResponsibleFor* and *Does*, which are relationships between roles and sets of activities. If a role is *ResponsibleFor* an activity, then either it does it directly, or it *Delegates* responsibility for it to another role. The Activity Hierarchy is then composed of a hierarchy of roles where the higher role is responsible for a superset of the activities of the lower role



(a) Role Hierarchy **(b) Generalisation**

Figure 1 A Role Hierarchy Based on Generalisation

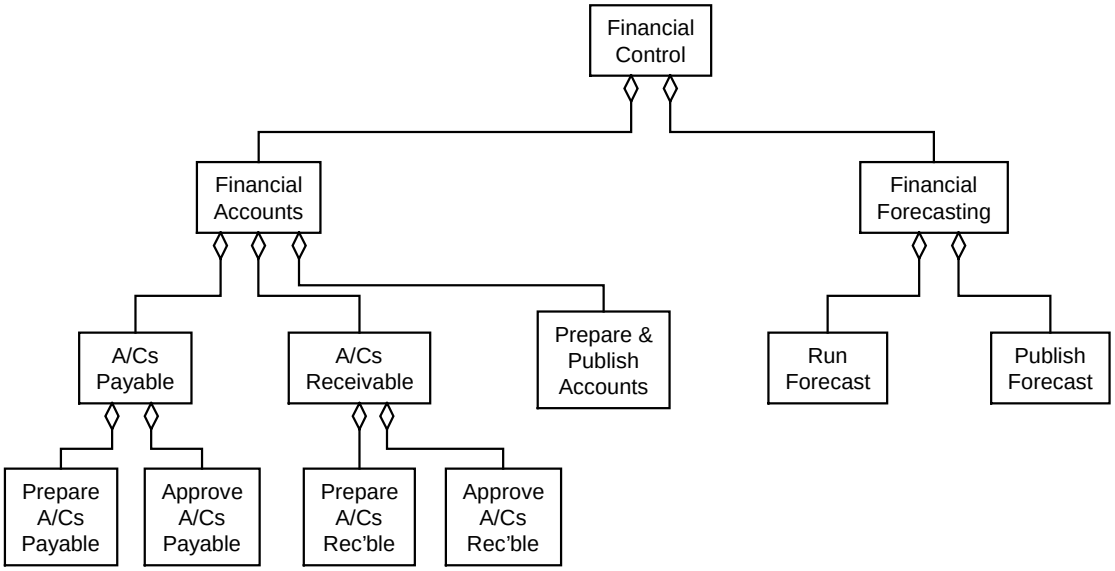


Figure 2 A Hierarchy Based on Aggregation

In the example that we are using, the activity hierarchy is identical with the organisational hierarchy (see figure 3), but there is no general reason why this should be so; responsibility may be delegated out to a different part of the organisation or contracted out.

2.3 Supervision Hierarchy

Most formal organisations describe their fixed positions by means of an organisation chart, which describes a hierarchy of named positions. An example is shown in figure 3. Each position has one or more roles:

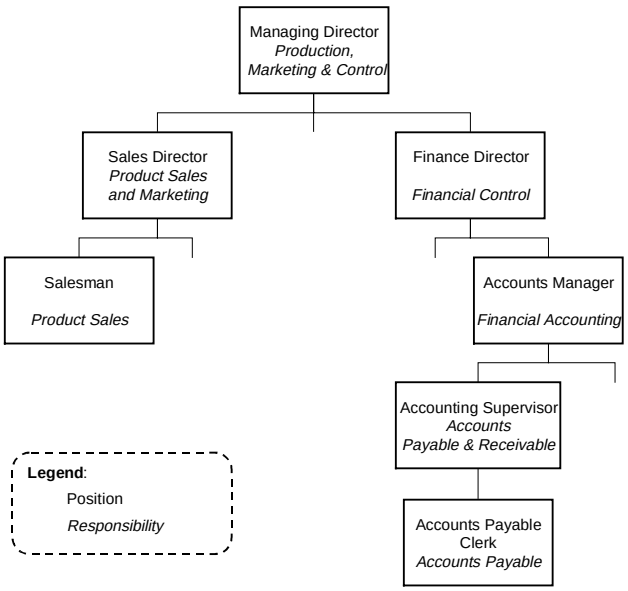


Figure 3 An Organisation Chart

- The Supervisor role for immediate inferiors in the hierarchy – it is this relationship which usually defines the hierarchy in the first place;
- The set of activities for which the position is responsible, shown in the figure in italics, e.g., the Finance Director has the role of Financial Control;
- The Review role for activities which this position is required to review. We mention this role because of its relevance to control principles, but it does not form the basis of a hierarchy; it is often, but not necessarily, carried out by the immediate superior in the hierarchy.

The supervision hierarchy is derived from the organisation chart, and is the hierarchy that tends to be assumed when we talk about an organisational hierarchy. It is, of course, a hierarchy of positions, not of roles.

Our reason for distinguishing between role and position is that the Supervisor role is inextricably bound up with the position, whereas the Activities and Review roles may be reassigned to other positions.

3 Organisational Control Principles

We summarise here the control principles which are typically used in large organisations. They were described in more detail in [4].

Separation of duties. This control principle is familiar to the computer security community from the Clark-Wilson commercial security model [6]. It is normally defined for critical transactions and is implemented by breaking the transaction into at least two separate actions. It is then

required that the two actions should not be performed by the same person. This is easily implemented in role-based access control by implementing role occupancy constraints and constraints on the inheritance of access rights within a role hierarchy. Users can then be prevented from occupying mutually incompatible roles, either simultaneously or in some time-related fashion [1]. Positive access rights for each of the actions are assigned only to the two incompatible roles, and the constraint enforces separation of duties.

Decentralisation or delegation. This control principle recognises that, in a large organisation, it is impossible for one person to manage directly all the activities of the organisation. Therefore, some activities are **delegated** by a "delegator" to other people who we will refer to as "delegates" (noun). They then have full authority to carry out those activities, though they are normally subject to supervision and review from their delegator. By delegating authority to the delegate, delegators abrogate their own immediate power to carry out those activities while retaining the right to revoke the delegation.

Supervision and review. There is of course a danger that delegates will not carry out their duties properly. For decentralisation to work satisfactorily, an additional control principle is needed: supervision and review. This control principle requires one person's actions to be reviewed post hoc by another person, typically their superior in the position hierarchy. The superior usually does not exert direct control over the supervisee at the time that the actions are taken.

- **Supervision** is an activity that is carried out on someone by the person in the immediately superior position. It consists of many activities including monitoring, appraisal, advising, praising and admonishing, and is outside the scope of any present-day access control system.

- **Review**, on the other hand, is carried out on specific activities. These activities can be controlled by an access control system provided that review is carried out as part of a computerised application.

Organisational control principles rarely figure in the everyday conversations of Computer Science academics, even those working in the field of security, and they have not received very much attention. However, the authors' experiences of large organisations make it clear that they all have explicit control principles which are in active use and enforced by auditors.

4 Authority State

The current state of an access control system is known as its **authority state**, which defines the result of any possible access request which may currently occur. It is derived from all the relevant facts about a system's state, present and past, which will determine whether an individual access request will be allowed or prevented. This includes (but is not limited to): system objects and agents, global system state, fixed access policies, and variable access rules.

4.1 Static Authority State

A great deal of work has been done on permissible values of the authority state and there is extensive literature on this area. Most work is concerned with specific, rather than general, policies, e.g. military security policies and the Chinese Wall Security Policy [7]. There has been recent work, however, which attempts to permit description of more general policies about the authority state. Jajodia and colleagues [8] have proposed a language for expressing authority state which is independent of individual security policies and can act as an integrating framework for multiple policies. Informally, the language offers expressions for access attempts, (discretionary) access rules, constraints upon those rules and verdicts upon access attempts. The constraints and verdicts can be used to express one or more security policies. This work offers a rather general approach to the description of permissible authority states, but deals with them entirely statically.

4.2 Changes to Authority State

However, for any access control policy which is described in terms of the static authority state, the set of potential states is much wider than the set of states that are actually acceptable to the organisation. This is true for Mandatory Access Control (MAC) policies just as much as for Discretionary Access Control (DAC) policies. For example a Security Officer may have wide discretion to re-label an object in a MAC system or make a new access rule in a DAC system, so far as the computer access control system is concerned. There are many actions, which are permitted under static security policies, but which are contrary to the organisation's policies. That is why such a high proportion of computer misuse incidents are attributed to "abuse of authority". It would be desirable to reduce the number of possible actions which constitute this abuse.

One contribution to ensuring that the authority state accurately reflects the organisation's security policies is to place constraints on permissible changes to the authority state by those who have authority to make these changes,

e.g. administrative roles. Changes must be constrained in terms of a number of factors, including who can change, and in what circumstances, user-role relationships, role permissions, and role relationships. Moreover, there is a need for a clear specification of what changes Security Officers can make e.g., which permissions they can add to a role.

Placing constraints on the change process will not eliminate all changes which are contrary to an organisation's policies, but inclusion of process considerations into the evaluation is a further step towards a high quality authority state. Most organisations do in fact include process constraints as part of the process of making changes to the authority state. However, they are typically outside the scope of access control models and computerised access control systems, being implemented by manual or informal procedures.

Clearly, changes to the authority state cannot be modelled by a static model of it, and it is impossible to verify, within a static access control model, that the change process was carried out in accordance with the organisation's policies. Consequently inspection of the new state cannot verify that it is legitimate, although it may be able to falsify it. Access control models need to be extended so as to include constraints upon changes to the authority state, i.e. a *dynamic* access control model is needed. There has been not been very much previous work which describes policies to achieve this. Two approaches have been the Grant-Take Model, and a hierarchical approach which is described below in section 6.2.

The Grant-Take Model is the oldest one which explicitly deals with authority state changes in computer systems. Its basis is that the authority to *change* access rights is tightly bound in with the right to *perform* an access; a user who has an access right may in addition have the right to **grant** that right to further users. The problem of how to trace the possible consequences of cascaded grants was first discussed by Saltzer [9]. Unfortunately, the Grant-Take paradigm tends to be in conflict with organisational control principles, particularly the principle of delegation, described in section 6.1.

5 Role Hierarchies & Static Authority State

In this section we discuss possible legitimate uses of role inheritance in the context of static authority state. We concentrate on making positive suggestions as far as possible, as we believe that the criticisms of unconstrained uses of role inheritance, which were made in our previous paper [4], have been generally accepted.

5.1 Read Access

The control principles that we have described have been concerned with maintaining the *integrity* of commercial transactions. When it comes to *confidentiality*, the integrity control principles do not apply. As a general principle, it could be organisational policy that anyone in the supervision hierarchy should be able to read any document which can be read by their inferiors.

The main problem with such a policy is that, without careful implementation, it could interfere with the integrity of documents if applied without thought. There needs to be protection against a superior reading a document that is in preparation, and making a copy before it has reached completion, which would effectively cause the integrity of the document to be violated. This objection can be dealt with in a principled fashion by introducing an access control policy that Read access rights inherit upwards, provided that the document's status is Completed, indicated by some suitable attribute value that is visible to the access control system.

5.2 Back-up Roles

All occupants of roles need a back-up person to deal with unplanned absences due to sickness, business travel, etc. Some of these can be anticipated, but there is always the possibility that a crucial occupant of a role goes absent at no notice, at a time when it is not possible to achieve an orderly change of authority state. One approach to this is the "password in an envelope" (usually in the top left-hand drawer of the person's desk!) which enables another person to impersonate the absentee. The well-known disadvantages of this are the lack of accountability which then ensues, and the need to perform recovery actions after the absentee's return.

Although in the last resort an envelope in the desk may be necessary, there is the possibility of a more orderly approach to cover many situations. Upwards inheritance of access rights, limited to one level only, enables the absentee's immediate superior to carry out emergency actions. This inevitably breaches the principle of supervision, since the superior is effectively (but necessarily) interfering with the inferior's role. However, this can be compensated for by ensuring that review of the action is subsequently carried out, either by an independent person or by someone further up the hierarchy.

5.3 Organisational Styles

The organisational style which leads to the imposition of control principles such as we have outlined is near-

universal in well-established bureaucratic organisations which deal in valuable assets such as money. However, there may be other organisations for which access right inheritance through the supervision hierarchy is appropriate, e.g., expert-led organisations where the boss is very proactive and is not constrained by control principles.

5.4 Discussion of Static Access Right Inheritance

Access right inheritance between roles, as described in [1], reduces the number of permissions within the system. However, the resulting hierarchy does not correspond to a conceptual relationship between the roles of an organisation and in particular it does not correspond to the supervision hierarchy on which most organisations are based. In order to avoid access right inheritance in the undesirable cases Sandhu [1] proposes the use of *private* roles e.g., Nurse' in figure 4. The users are then assigned to the private role Nurse' and the Nurse role becomes virtual.

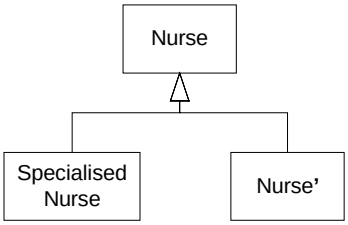


Figure 4 Private roles

This concept is further extended to define private role hierarchies where inheritance relationships are exhibited between private roles. However, in an organisation where users may hold private files on the company's computers, or in the case of draft documents to be protected from access by superiors, this technique would require assigning every user in the system to a private role. Thus, the benefits of a reduction in the number of permissions, obtained by the use of inheritance, are counter-balanced by an increased number of roles and a more complex role hierarchy specification.

The possible uses of access right inheritance that we have described in the previous sections, considering only a static authority state, have little in common with the examples given in the original RBAC paper [1]. It was perhaps natural to transplant the inheritance properties of the object-oriented paradigm and assume that, in access control also, properties are inherited up the hierarchy. However, there are two problems with this approach:

- Inheritance up a hierarchy can lead to the violation of organisational control principles, as was recognised in [4];
- In any case the most typical role hierarchy of an organisation is not the *isa* hierarchy but the supervision hierarchy, and the object-oriented inheritance paradigm is not appropriate to this hierarchy.

The uses of static access right inheritance that we have illustrated in this section appear to us to be rather unimportant. Bearing in mind the necessity for exceptions to the first two, we doubt whether it is worthwhile introducing the additional complication of an access rights inheritance mechanism solely for these purposes.

6 Role Hierarchies & Changes to Authority State

By contrast with our rather pessimistic conclusions about static inheritance of access rights, we see role hierarchies, based on the natural hierarchies of an organisation, as being crucial to the effective control of changes to the authority state of an organisation. Because "inheritance" has acquired static connotations in computer science we will use the term "propagation" to denote the dynamic transmission of properties from one entity to another, as a result of actions that are performed.

6.1 Delegation

It is our view that the principle of delegation is crucial to the success of large enterprises, whether they are formal organisations or looser collections of co-operating individuals, such as the Internet or Linux communities. We regard these loose collections as being of great importance and interest, but the subject matter of this paper is hierarchies, and we will concentrate on formal organisations with a hierarchical structure.

Recapping on the summary in section 3, delegates have full authority to carry out activities which have been delegated to them. Delegators abrogate their own immediate power to carry out those activities for two reasons: there would be no reason to delegate if the delegator could carry them out satisfactorily; and once having delegated an activity, the appropriate way of ensuring that it is carried out satisfactorily is by supervision and review, not by direct action.

As a result of an act of delegation, the delegate receives responsibility for that activity and the delegator is no longer directly responsible for it. Note two important points:

- The delegator, although no longer directly responsible for the action, is usually still responsible, indirectly, for ensuring that it is carried out;

- The act of delegation, though it may relieve the delegator of the direct duty to carry out the activity, does not usually relieve that delegator of accountability. If serious consequences follow from the faulty carrying out of a delegated activity, the delegator cannot usually avoid blame by claiming that responsibility was delegated.

Given the principle of delegation, the following rules are required:

- Delegators can only delegate activities for which they are responsible;
- After an activity has been delegated it cannot be:
  - Performed by the delegator, or
  - Delegated again

  unless the first delegation is revoked.

6.2 Delegation Hierarchies

In a formal organisation, delegation is a natural means of decentralising control. It is the way in which organisations work in practice. When a new area of activity (a role) is started, a designated person is delegated the responsibility to carry out the role and given the authority and resources that are needed for the purpose. Some of the activities in the role are carried out directly by the person, and other are delegated again. Delegation may take place in one of two ways:

- Without reference to the existing supervision hierarchy, in which case a natural activity hierarchy is induced by the process of splitting the role into subsets and delegating those subsets downwards;
- Within an existing organisation, with the activity being delegated down the existing supervision hierarchy.

6.2.1 Propagation through the Activity Hierarchy

We described the Activity Hierarchy in section 2.2. It is defined by the aggregation relationship; informally, the roles further down the hierarchy are responsible for a subset of the activities of roles which are higher up the hierarchy. Let us assume that person P has been given a role R1. Then the access control system is assumed to give P permission for all actions in R1 and also permission to Delegate(R1, *x*, *grant*) where *x* is a person and *grant* is True or False. Then, if R2 is a subset of R1 and Q is a person:

- P has permission to perform action Delegate(R2, Q, *grant*). After it has been performed, then
- Q has permission for all actions in R2 and also, if *grant* is True, permission to Delegate(R2, x, *grant*);
- P now has permission for all actions in (R1 – R2) and also permission to Delegate((R1 – R2), x, *grant*);
- P also has permission to Revoke(R2, Q).

Thus actions can be delegated as far down an Activity Hierarchy as is wished, and the access control system enforces the rules described in section 6.1.

This hierarchy has a lot in common with the Grant-Take Model, with its associated problems of tracing the consequences of grants, and revocation. The problem of how to trace the possible consequences of cascaded grants was discussed by Saltzer [9]. The problem of how the original owner can revoke cascaded grants has been discussed by Griffiths and Wade [10], whose revocation algorithm has subsequently been improved by a variety of authors, most recently [11].

6.2.2 Propagation through the Supervision Hierarchy

We described the Supervision Hierarchy in section 2.3. It is defined by means of an organisation chart, which describes a hierarchy of named positions. A more conservative approach than the scheme of section 6.2.1 is to restrict the scheme with a further condition, based on the Supervision Hierarchy:

- The action Delegate(R2, Q, *grant*) is only permitted to P if P is the (immediate) superior of Q in the Supervision Hierarchy.

The effect of this will be to restrict P from contracting out responsibilities outside P's part of the organisation. The additional stipulation "immediate" might or might not be enforced depending upon organisational style.

Schemes of this kind, but less general, have been described by [12] and developed by [13], with extensions designed to solve the problem of separation of duties for Security Administrators. They enable an access control system to permit a Security Administrator to grant authority without being permitted to exercise that same authority. Hierarchical delegation schemes are demanded by large commercial organisations, and have been implemented in *ad hoc* fashion by commercial mainframe access control systems such as RACF and ACF2.

6.3 Discussion of Dynamic Propagation Access Rights

Unlike static inheritance, the dynamic propagation of access rights through controlled changes to the authority state has been demanded by commercial organisations and provided, in one form or another, for a number of years.

However, this area has been relatively neglected by computer scientists.

The difficulty, for any formal proposal, is in deciding how much should be integrated into the access control system, and how much should be enforced by the implementation of *ad hoc* restrictions which are added to the access control system through wrappers or user-coded procedures. There has been no systematic attempt that we are aware of up to now to investigate the need for, and the implementation of, a canonical set of constraints on changes to the authority state. At the time of writing, projects are due to start shortly at both the authors' Universities which will make some progress in this area.

7 Conclusions and Further Work

The conclusion of this discussion is that the focus of research on hierarchies in access control needs to be changed from static inheritance of access rights through role hierarchies, to the use of hierarchies in providing constraints upon changes to authority state. This needs to be approached with a two-pronged attack:

- Formal modelling of permissible authority state transitions, in order to enable prediction of the consequences of particular policies;
- Examination of the real requirements of large organisations, so that the eventual proposals can be tested against real requirements.

We hope to be able to report progress on both these fronts in due course.

References

1. Sandhu, R.S., *et al., Role-Based Access Control Models.* IEEE Computer, 1996. **29**(2): p. 38-48.
2. Lupu, E.C. and M.S. Sloman. *Reconciling Role-Based Management and Role-Based Access Control.* in *2nd ACM Workshop on Role-Based Access Control*. 1997. George Mason University, Fairfax, Virginia, USA.
3. Awischus, R. *Role Based Access Control with the Security Administration Manager (SAM).* in *2nd ACM Workshop on Role-Based Access Control.* 1997. George Mason University, Fairfax, Virginia, USA.
4. Moffett, J.D. *Control Principles and Role Hierarchies*. in *3rd ACM Workshop on Role Based Access Control (RBAC).* 1998. George Mason University, Fairfax, VA.
5. Thomas, E. and B. Biddle, *The Nature and History of Role Theory*, in *Role Theory: Concepts and Research*, B. Biddle and E. Thomas, Editors. 1979, Krieger Publishing.
6. Clark, D.C. and D.R. Wilson. *A Comparison of Commercial and Military Computer Security Policies*. in *IEEE Symposium on Security and Privacy*. 1987. Oakland, CA: IEEE Computer Society Press.
7. Brewer, D.F.C. and M.J. Nash. *The Chinese Wall Security Policy*. in *IEEE Symposium on Security and Privacy*. 1989. Oakland, CA: IEEE Computer Society Press.
8. Jajodia, S., P. Samarati, and V.S. Subrahmanian. *A Logical Language for Expressing Authorizations*. in *IEEE Symposium on Security and Privacy*. 1997. Oakland, CA: IEEE Computer Society Press.
9. Saltzer, J.H. and M.D. Schroeder, *The Protection of Information in Computer Systems.* Proceedings of the IEEE, 1974. **63**(9): p. 1278-1303.
10. Griffiths, P.P. and B.W. Wade, *An Authorization Mechanism for a Relational Database System.* ACM Transactions on Database Systems, 1976. **1**(3): p. 242-255.
11. Jonscher, D., *Access Control in Object-Oriented Federated Database Systems*. 1998: Infix. ISBN 3 89601 449 8.
12. Rotenberg, L.J., *Making Computers Keep Secrets.* 1974. MIT, MAC TR-115, Feb 1974.
13. Moffett, J.D. and M.S. Sloman, *Delegation of Authority*, in *Integrated Network Management II*, I. Krishnan and W. Zimmer, Editors. 1991, North Holland. p. 595-606.