Administrative Scope and Role Hierarchy Operations

Jason Crampton and George Loizou
School of Computer Science and Information Systems
Birkbeck College, University of London

ABSTRACT

The ARBAC97 model makes an important contribution to the understanding and modeling of administration in role-based access control. However, there are several features of the model which we believe could be improved. We introduce the concept of administrative scope in a role hierarchy and show how this can be used to control updates to the hierarchy. We then incrementally develop a model for administering the role hierarchy and compare it to the RRA97 sub-model of ARBAC97. We conclude that our model offers significant advantages over RRA97.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*Access controls*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection; I.6.0 [**Computing Methodologies**]: Simulation and Modeling

General Terms

Security, Theory

Keywords

role-based access control, administrative scope, encapsulated range, role hierarchy operation

1. INTRODUCTION

Role-based access control (RBAC) models have been the subject of considerable research in recent years resulting in several important models: the NIST model [3]; the role graph model [6]; the RBAC96 model [8] and the recent unified NIST RBAC model [9]. It has been suggested that such models provide an attractive theoretical framework for multi-domain, distributed systems [5]. The features that make RBAC attractive include policy neutrality, principle of least privilege and ease of management. Gligor [4] provides a good introduction to the characteristics and advantages of RBAC. The material in this paper is developed in the context of the RBAC96 model. In particular, we assume the existence of a partially ordered set of roles R (which is visualized as a role hierarchy).

The use of RBAC principles to manage RBAC systems has been less widely studied although significant advances have been made. The NIST model and implementation of RBAC incorporates an Admin Tool which provides administrative support for an RBAC database which stores information about user-role, permission-role assignments and the role hierarchy structure [3]. The role graph model includes several algorithms for manipulating the role graph in order to support administrative functions [6]. ARBAC97 [7], the most elaborate of these attempts, provides a complete model for administration in the context of the RBAC96 model. ARBAC97 supports decentralized administration and incorporates the functionality provided by the NIST and role graph models.

Therefore, in this paper we will examine the ARBAC97 model in more detail and explain why we believe that an alternative approach to administration is required. Our approach is inspired in part by the notion of an *encapsulated range* which plays an integral part in the administration of the role hierarchy in ARBAC97. However, the development of our role hierarchy administration model (RHA) is distinctly different from that of ARBAC97. We believe that RHA is a more robust, flexible, widely applicable and less complex model than ARBAC97.

In fact, RHA only considers the administration of the role hierarchy. Therefore, we are actually proposing an alternative to RRA97, the sub-model of ARBAC97 concerned with administration of the role hierarchy. We have developed a complete model for RBAC administration which will be the subject of a future paper.

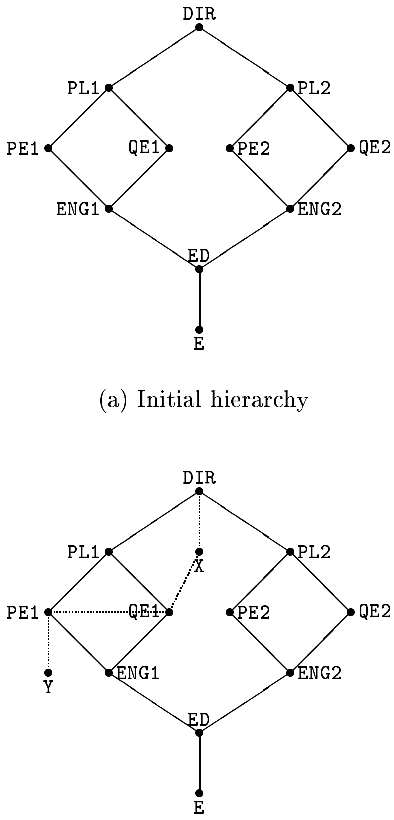
In the next section we discuss the RRA97 model in more detail and explain some of its disadvantages. In Section 3 we introduce the notion of *administrative scope* which is the

SACMAT'02, June 3-4, 2002, Monterey, California, USA.
Copyright 2002 ACM 1-58113-496-7/02/0006 ...$5.00.

fundamental concept in RHA. In Section 4 we demonstrate how administrative scope is used to control changes to the role hierarchy and also introduce a family of increasingly complex administrative models RHA_1–RHA_4. In the penultimate section we compare RHA_4 with RRA97. Finally, we summarize our contribution and discuss future work.

2. RRA97 AND ENCAPSULATED RANGES

This section is not intended to be a comprehensive description of ARBAC97. Rather, we will review the definition of an encapsulated range and its purpose in the context of RRA97. Figure 1a shows a role hierarchy that has been used as an example in earlier papers by Sandhu [7]. We will use this hierarchy to illustrate both the motivation for and the shortcomings of RRA97.



(a) Initial hierarchy

(b) Undesirable changes

Figure 1: An RBAC96 role hierarchy and the effect of undesirable changes

The hierarchy shown in Figure 1a models the roles in an Engineering Department. For example, `PE1` and `PE2` are production engineer roles; `PL1` and `PL2` are project leader roles. RRA97 considers the following (*role hierarchy*) *operations*: role insertion, role deletion, edge insertion and edge deletion. We will denote these by `AddRole`$(r, \Delta r, \nabla r)$, `DeleteRole`(r), `AddEdge`(c, p) and `DeleteEdge`(c, p), respectively, where Δr is the set of immediate children of r, ∇r is the set of immediate parents of r, c the child role and p the parent role. We assume that an operation does not introduce a cycle into the hierarchy. Specifically, $p \not< c$ and for all $s \in \Delta r$ and all $t \in \nabla r$, $s \not> t$.

Now consider the following sequence of operations:

- `AddRole`$(\texttt{X}, \{\texttt{QE1}\}, \{\texttt{DIR}\})$ – add the role `X` and edges $(\texttt{QE1}, \texttt{X})$ and $(\texttt{X}, \texttt{DIR})$;
- `AddRole`$(\texttt{Y}, \emptyset, \{\texttt{PE1}\})$ – add the role `Y` and the edge $(\texttt{Y}, \texttt{PE1})$;
- `AddEdge`$(\texttt{PE1}, \texttt{QE1})$ – add the edge $(\texttt{PE1}, \texttt{QE1})$.

The cumulative effect of these three operations is to make $\texttt{Y} < \texttt{X}$; this is illustrated in Figure 1b and is considered to be an "anomalous side effect" [7] of unconstrained changes to the role hierarchy. Therefore, RRA97 seeks to provide a framework in which such side effects cannot occur. (It is unclear to us why these side effects should be considered anomalous. For example, in Figure 1a, $\texttt{PE1} \not< \texttt{QE1}$, but RRA97 permits the operation `AddEdge`$(\texttt{PE1}, \texttt{QE1})$ which causes `PE1` to become junior to `QE1`. In short, there seems to be no qualitative difference between `Y` being made more junior to `X` and `PE1` being made more junior to `QE1`.) RRA97 also seeks to "maximize the potential for decentralization of administration and autonomy of administrative roles" [7].

2.1 The essential components of RRA97

The fundamental idea in RRA97 is that of an encapsulated range. (Recall that an *open* range (x, y) is defined to be the set $\{r \in R : x < r < y\}$. Similarly, a *closed* range $[x, y]$ is defined to be the set $\{r \in R : x \leqslant r \leqslant y\}$.) The following definition is due to Sandhu *et al.* [7].

Definition 2.1. *A range (x, y) is said to be* encapsulated *if for all $w \in (x, y)$, and for all $z \notin (x, y)$,*

$$z > w \text{ if, and only if, } z > y, \text{ and} \tag{1}$$

$$z < w \text{ if, and only if, } z < x. \tag{2}$$

Informally, an encapsulated range is a self-contained subhierarchy in the role hierarchy with all external edges passing through one of the end points of the range. $(\texttt{E}, \texttt{ED})$, $(\texttt{ENG1}, \texttt{PL1})$ and $(\texttt{ED}, \texttt{DIR})$ are examples of encapsulated ranges in Figure 1a.

The `can-modify` $\subseteq AR \times \mathcal{E}(R)$ relation, where AR is the set of administrative roles and $\mathcal{E}(R)$ is the set of encapsulated ranges in R, determines the encapsulated ranges over which administrative roles can act. Table 1 shows a typical example of the `can-modify` relation [7].[1] An encapsulated

[1] `PSO1` denotes project 1 security officer role; `DSO` denotes departmental security officer role.

range that appears in the `can-modify` relation is called an *authority range*. RRA97 also requires that for any two authority ranges, they either be disjoint or one be entirely contained in the other.

Hence, for every role $r \in R$, there is a unique smallest authority range to which r belongs. This is called the *immediate authority range* of r. For example, given the `can-modify` relation in Table 1, the immediate authority range of PE1 is $(\texttt{ENG1}, \texttt{PL1})$, not $(\texttt{ED}, \texttt{DIR})$, while the immediate authority range of PE2 is $(\texttt{ED}, \texttt{DIR})$.

can-modify	
Administrative Role	Authority Range
PSO1	(ENG1, PL1)
DSO	(ED, DIR)

Table 1: The `can-modify` relation

A range (x, y) is a *create range* if one of the following conditions is satisfied:

- x and y have the same immediate authority range;
- y is the (upper) end point of the immediate authority range of x;
- x is the (lower) end point of the immediate authority range of y.

In RRA97, an administrative role $a \in AR$ such that $(a, (w, z)) \in \texttt{can-modify}$ can perform the operation:

- $\texttt{AddRole}(r, \Delta r, \nabla r)$ if $\Delta r = \{x\}$, $\nabla r = \{y\}$, (x, y) is a create range, $w \leqslant x$ and $y \leqslant z$;
- $\texttt{DeleteRole}(r)$ if $r \in (w, z)$;
- $\texttt{AddEdge}(c, p)$ if $w \leqslant c$, $p \leqslant z$ and either
  - the immediate authority range of c equals the immediate authority range of p; or
  - there exists an authority range (u, v) such that, either $c = u$ and $p < v$ or $c > u$ and $p = v$, and the insertion of (c, p) does not violate the encapsulation of (u, v);
- $\texttt{DeleteEdge}(c, p)$ if (c, p) is not an authority range and $w \leqslant c$ and $p \leqslant z$.

It can be seen that the requirements for the `AddRole` and `AddEdge` hierarchy operations to succeed are rather complicated. Furthermore, in the case of `AddRole`, the insertion of a role which has no parent or child or which has more that one parent or child is not permitted. In particular, $\texttt{AddRole}(\texttt{Y}, \emptyset, \{\texttt{PE1}\})$ is not permitted. $\texttt{AddRole}(\texttt{X}, \{\texttt{QE1}\}, \{\texttt{DIR}\})$ is not permitted because $(\texttt{QE1}, \texttt{DIR})$ is not a create range. $\texttt{AddEdge}(\texttt{PE1}, \texttt{QE1})$ is permitted.

Remark 2.1. *We note that Definition 2.1 implies no range can be encapsulated since $y \notin (x, y)$, $y > w$ for all $w \in (x, y)$ but $y \not> y$. Hence conditions* (1) *and* (2) *should be replaced by*

$$z > w \text{ if, and only if, } z \geqslant y \text{ and} \tag{3}$$

$$z < w \text{ if, and only if, } z \leqslant x, \tag{4}$$

respectively.

3. ADMINISTRATIVE SCOPE

We believe that the RRA97 model has several shortcomings: it suffers from a lack of applicability, flexibility, coherence and robustness; its interaction with the other submodels of ARBAC97 is not completely determined; it is rather complex and lacks intuitive appeal [1]. We believe that many of these problems arise because of two particular features in the development of the ARBAC97 model.

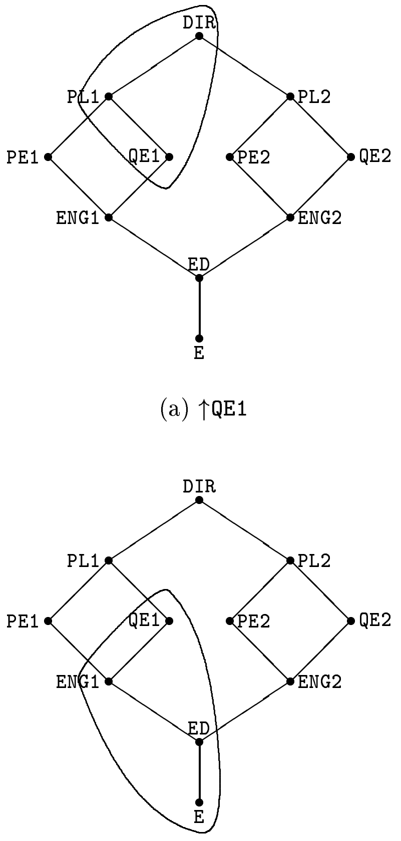
Firstly, we believe that a sensible approach to the problem of administration in role-based access control is to first determine how hierarchy operations are to be performed. However, in ARBAC97, the “easy” models, URA97 and PRA97 (which deal with user-role and permission-role assignment, respectively), were developed first; as a result of this, the integration of these models with RRA97 has not been easy to achieve [1].

Secondly, the development of RRA97 was based on encapsulated ranges. The reason for this decision was that the model should support decentralization, autonomy and should not allow anomalous changes to the hierarchy. It is clear that the decision to develop an administrative model that requires the existence and preservation of encapsulated ranges in a role hierarchy has a significant and detrimental effect on the applicability of the resulting model. In other words, the development of ARBAC97 has been driven by the concept of an encapsulated range not by the needs of RBAC96; surely this is the wrong way round. Furthermore, although RRA97 guarantees that changes are local and cannot propagate undesirable changes through the hierarchy, it is not obvious that the concept of an encapsulated range is the most appropriate basis for developing an administrative model.

We believe that the requirements for ARBAC97 missed several important points. For example, surely RRA97 should be applicable to as many role hierarchies as possible. We believe it is more appropriate to develop a more permissive model that does not preclude administrative changes because they conflict with the assumptions of the administrative model. Rather, the model should permit changes if they are reasonable in some intuitive sense. (This requirement is no more vague than the requirement that RRA97 should not permit anomalous changes.) Clearly, such a model should provide support for accountability and should be easily integrated with RBAC96.

In this section we will define the concept of administrative scope and show how this can be used to control hierarchy operations. The idea is inspired by an alternative formulation of encapsulated range which we present in Proposition 3.1.

We first introduce some notation borrowed from partial order theory [2]. Let $S \subseteq R$; define $\uparrow S = \{r \in R : r \geqslant s \text{ for some } s \in S\}$ and $\downarrow S = \{r \in R : r \leqslant s \text{ for some } s \in S\}$. If $S = \{s\}$ we will simply write $\uparrow s$ and $\downarrow s$. Figure 2 shows $\uparrow$QE1 and $\downarrow$QE1; relevant roles lie within a closed curve.



(a) $\uparrow$QE1

(b) $\downarrow$QE1

Figure 2: $\uparrow$QE1 and $\downarrow$QE1

Proposition 3.1. *A range (x, y) is encapsulated if, and only if,*

$$\uparrow(x, y) \setminus \uparrow y = (x, y) \text{ and} \tag{5}$$

$$\downarrow(x, y) \setminus \downarrow x = (x, y). \tag{6}$$

Proof. This proof assumes the characterization of encapsulated range given in Remark 2.1.

$\Rightarrow$ Suppose for all $z \notin (x, y)$ and for all $w \in (x, y)$ we have $z > w$ if, and only if, $z \geqslant y$. We now prove that $\uparrow(x, y) \setminus \uparrow y \subseteq (x, y)$. Let $a \in \uparrow(x, y) \setminus \uparrow y$. Then there exists $b \in (x, y)$ such that $b \leqslant a$ and $y \nleqslant a$. Since (x, y) is encapsulated, $a \in (x, y)$ (otherwise we have $a \notin (x, y)$ such that $a \geqslant b$ for some $b \in (x, y)$ and $y \ngeqslant a$). Clearly, $(x, y) \subseteq \uparrow(x, y) \setminus \uparrow y$ and hence we have $\uparrow(x, y) \setminus \uparrow y = (x, y)$.

The corresponding proof for $\downarrow(x, y) \setminus \downarrow y$ is similar; we omit the details.

$\Leftarrow$ Suppose $\uparrow(x, y) \setminus \uparrow y = (x, y)$. Let $w \in (x, y)$ and $z \notin (x, y)$ with $z > w$. Hence $z \in \uparrow(x, y)$. Since $z \notin (x, y)$, $z \in \uparrow y$ and hence $z \geqslant y$.

The corresponding proof for $\downarrow(x, y) \setminus \downarrow y$ is similar; we omit the details.

□

Our model is motivated by the following two intuitively reasonable suggestions for resolving the problems posed by the hierarchy operations that led to Figure 1b. Namely, once role X has been created:

- Remove QE1 from PSO1's administrative range as QE1 is now less than X, a role which is not in PSO1's administrative range. That is, only DSO and above should now be able to administer QE1. In particular, PSO1 would not be able to make PE1 less than QE1.
- A role r such that $|\nabla r| > 1$ (such as QE1 once X has been inserted into the hierarchy) must be administered by a role which has administrative control over every role in ∇r. In our example, only DSO would be able to make PE1 less than QE1.

These solutions have a similar approach and could be implemented by imposing upper limits on the authority of each administrative role. Therefore, we define administrative scope to model this behaviour.

Definition 3.1. *The* administrative scope *of a role r is defined as follows:*

$$\mathcal{A}(r) = \{s \in R : s \leqslant r, \uparrow s \setminus \uparrow r \subseteq \downarrow r\}. \tag{7}$$

For example, in Figure 1a, $\text{ENG1} \in \mathcal{A}(\text{PL1})$ because $\uparrow\text{ENG1} = \{\text{ENG1}, \text{PE1}, \text{QE1}, \text{PL1}, \text{DIR}\}$ and $\uparrow\text{PL1} = \{\text{PL1}, \text{DIR}\}$; hence $\uparrow\text{ENG1} \setminus \uparrow\text{PL1} = \{\text{ENG1}, \text{PE1}, \text{QE1}\} \subset \downarrow\text{PL1}$. It can easily be seen that $\mathcal{A}(\text{PL1}) = \{\text{PL1}, \text{PE1}, \text{QE1}, \text{ENG1}\}$. However, $\text{ENG1} \notin \mathcal{A}(\text{PE1})$, for example, because $\text{QE1} \in \uparrow\text{ENG1}$ and $\text{QE1} \notin \downarrow\text{PE1}$.

Informally, administrative scope has characteristics similar to those exhibited at the upper end point of an encapsulated range. That is, there is only one way into the administrative scope of r from above and that is through r itself. More formally, we have the following proposition which shows that administrative scope is a less restrictive notion than range encapsulation.

PROPOSITION 3.2. *If (x, y) is an authority range, then $(x, y) \subseteq \mathcal{A}(y)$.*

PROOF. Suppose $z \in (x, y)$. Then $x < z < y$ and hence $\uparrow x \supset \uparrow z \supset \uparrow y$. Therefore,

$$\begin{aligned} \uparrow z \setminus \uparrow y &\subset \uparrow(x, y) \setminus \uparrow y \\ &= (x, y) \quad \text{by (5)} \\ &\subset \downarrow y. \end{aligned}$$

That is, $z \in \mathcal{A}(y)$. □

3.1 Flexibility of administrative scope

The administrative scope of a role is determined by the role hierarchy and changes dynamically as the hierarchy changes. (This is in contrast to RRA97, where administration is largely determined by the `can-modify` relation, which in turn imposes restrictions on changes that can be made to the hierarchy.) For example, following the operation `AddRole`$(\texttt{X}, \{\texttt{QE1}\}, \{\texttt{DIR}\})$, `QE1` $\notin \mathcal{A}($`PL1`$)$. Figure 3 shows how the administrative scope of `PL1` changes as edges and roles are added to the hierarchy.

3.2 Decentralization and autonomy

It can be seen that for all $r \in R$, $r \in \mathcal{A}(r)$. Hence we define the *strict* administrative scope of r to be $\mathcal{A}(r) \setminus \{r\}$, which we will denote $\mathcal{A}_S(r)$. If $s \in \mathcal{A}_S(r)$ we say r is an *administrator* of s.

PROPOSITION 3.3. *If r has an administrator then the set of administrators of r has a unique* minimal *administrator which we refer to as the* line manager *of r.*

PROOF. If r has a single administrator the result follows immediately. Therefore, suppose x and y are minimal administrators of r. (That is, for all administrators z of r, $z \leqslant x$ implies $z = x$ and $z \leqslant y$ implies $z = y$. Hence, $x \not< y$ and $y \not< x$.) Then $r \in \mathcal{A}_S(x)$ and hence $x \in \uparrow r$. Similarly, $r \in \mathcal{A}_S(y)$ and hence by (7)

$$\uparrow r \setminus \uparrow y \subseteq \downarrow y. \tag{8}$$

Since $y \not< x$, $x \notin \uparrow y$ and hence $x \in \downarrow y$ by (8). Hence $x < y$, which is a contradiction. □

The concept of line manager can be applied to administration of the role hierarchy to ensure maximum decentralization and accountability. That is, we can insist that all changes affecting a role are made by the line manager. This feature could be particularly useful in the management of user-role and permission-role assignments.

4. A FAMILY OF MODELS FOR HIERARCHY ADMINISTRATION

In this section we describe a family of models for hierarchy administration of increasing sophistication (and incurring larger overheads). We will discuss the relative merits of each of these models in order to justify why we use a particular model for comparison with RRA97 in Section 5. In common with RRA97, we assume throughout that hierarchy operations are initiated by another ("administrative") role a. However, unlike in RRA97, we do not assume the existence of a disjoint set of administrative roles.

4.1 RHA$_1$

RHA$_1$ is the basic model and defines under what circumstances, defined in terms of administrative scope, a hierarchy operation succeeds. We permit a role a to perform the hierarchy operation:

- `AddRole`$(r, \Delta r, \nabla r)$ provided $\Delta r \subseteq \mathcal{A}_S(a)$, $\nabla r \subseteq \mathcal{A}(a)$ and $|\Delta r| + |\nabla r| > 0$ (r has at least one parent or child role);
- `DeleteRole`(r) provided $r \in \mathcal{A}_S(a)$;
- `AddEdge`(c, p) provided $c, p \in \mathcal{A}(a)$;
- `DeleteEdge`(c, p) provided $c, p \in \mathcal{A}(a)$.

Clearly RHA$_1$ has the benefit of great simplicity. It can be incorporated directly into RBAC96 without the need for any additional relations. Furthermore, it admits the decentralization of administration. For example, the project leader role `PL1` can administer the roles in project 1.

However, it is unlikely that RHA$_1$ will provide a sufficiently fine-grained approach to administration and security in many applications. For example, `E` $\in \mathcal{A}_S($`ED`$)$, but it is probably undesirable that `ED` should have any control over the hierarchy.

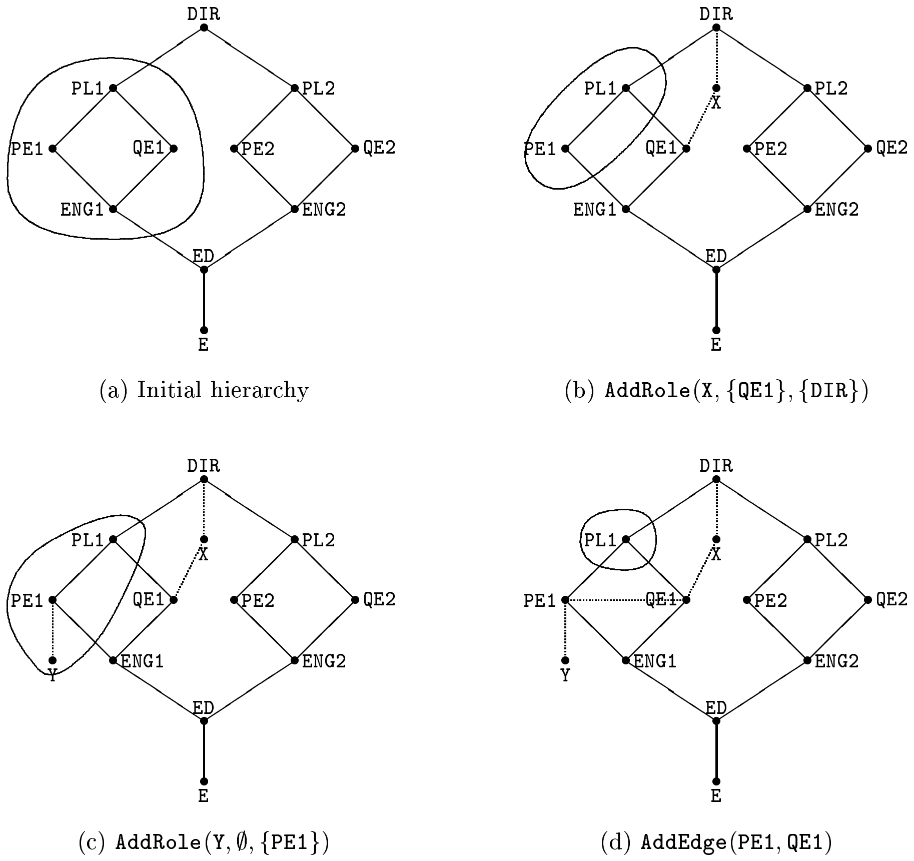
4.2 RHA$_2$

We can extend RHA$_1$ by insisting that, in addition to satisfying the administrative scope conditions, a must also have appropriate (administrative) permissions assigned to it in order to perform hierarchy operations. RHA$_2$ can be implemented without introducing additional relations and offers finer granularity than RHA$_1$ without incurring any significant overheads.

4.3 RHA$_3$

In this model we introduce a binary relation `admin-authority` $\subseteq R \times R$. If $(a, r) \in$ `admin-authority` then a is called an *administrative role*;[2] we also say a

[2] We observe that (a, r) could denote a range in the role



(a) Initial hierarchy

(b) `AddRole`$(\texttt{X}, \{\texttt{QE1}\}, \{\texttt{DIR}\})$

(c) `AddRole`$(\texttt{Y}, \emptyset, \{\texttt{PE1}\})$

(d) `AddEdge`$(\texttt{PE1}, \texttt{QE1})$

Figure 3: The dynamic nature of administrative scope: The roles inside the closed curve denote the administrative scope of PL1

controls r. We denote the set of roles that a controls by $C(a)$.

We first make the observation that the `admin-authority` induces an *extended* hierarchy on the set of roles which includes the original hierarchy. For example, the `admin-authority` relation defined in Figure 4a results in the extended hierarchy in Figure 4b. The elements of `admin-authority` are represented by broken lines. (All subsequent examples in this paper will be visualized using an extended hierarchy rather than explicitly defining the `admin-authority` relation.)

We extend the definition of administrative scope in a natural way: namely,

$$\mathcal{A}(a) = \{r \in R : \uparrow r \setminus \uparrow C(a) \subseteq \downarrow C(a)\} \quad \text{and}$$
$$\mathcal{A}_S(a) = \mathcal{A}(a) \setminus C(r),$$

where the evaluation of $\uparrow r$, $\uparrow C(a)$ and $\downarrow C(a)$ takes place in the extended hierarchy. For example, in Figure 5b, $C(\texttt{PSO1}) = \{\texttt{X}, \texttt{PE1}, \texttt{QE1}\}$ and $\mathcal{A}(\texttt{PSO1}) = \{\texttt{X}, \texttt{PE1}, \texttt{QE1}, \texttt{ENG1}\}$; furthermore, $\texttt{PSO1} \in \mathcal{A}(\texttt{DSO})$.

There are two self-evident consistency requirements that `admin-authority` must satisfy: for all $(a, r) \in$ `admin-authority`, $a \not\leqslant r$; and `admin-authority` is antisymmetric. In addition, we require that the second field in `admin-authority` be unique. In other words each $r \in R$ is controlled by at most one administrative role. This constraint is introduced in order to preserve the line manager feature of the preceding models.

We permit an administrative role a to perform the hierarchy operation:

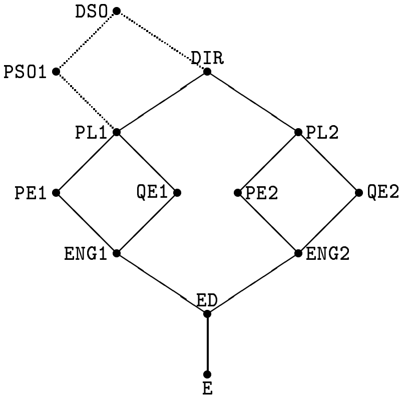
- `AddRole`$(r, \Delta r, \nabla r)$ provided $\Delta r \subseteq \mathcal{A}_S(a)$ and $\nabla r \subseteq \mathcal{A}(a)$;
- `DeleteRole`(r) provided $r \in \mathcal{A}_S(a)$;
- `AddEdge`(c, p) provided $c, p \in \mathcal{A}(a)$;
- `DeleteEdge`(c, p) provided $c, p \in \mathcal{A}(a)$.

RHA$_3$ provides a level of indirection not available in RHA$_1$ and RHA$_2$ and therefore can be used to implement a far

hierarchy, or an edge in the hierarchy or a tuple in the `admin-authority` relation. However, the interpretation of (a, r) will always be clear from context and the symbols chosen.

admin-authority	
Administrative Role	Role
PSO1	PL1
DSO	DIR
DSO	PSO1

(a) The admin-authority relation



(b) The extended hierarchy

Figure 4: An extended hierarchy

more flexible security policy. The admin-authority relation states which administrative roles have responsibility for which parts of the role hierarchy. In this sense, it is similar to the can-modify relation in RRA97. For example, the can-modify relation in Table 1 can be replaced by the admin-authority relation in Figure 4a. (($\mathtt{DSO}, \mathtt{PSO1}$) is included for the development and discussion of the model in Section 4.4.) Given this admin-authority relation, $\mathcal{A}(\mathtt{DSO}) = [\mathtt{E}, \mathtt{DIR}] \cup \{\mathtt{PSO1}\}$ and $\mathcal{A}(\mathtt{PSO1}) = [\mathtt{ENG1}, \mathtt{PL1}]$.[3]

Finally we note that RHA_1 is a special case of RHA_3, where $(r, r) \in$ admin-authority for all $r \in R$.

4.4 RHA$_4$

In this section we consider how RHA_3 can be extended to administer the admin-authority relation. We need to consider when and how the admin-authority relation can be updated by hierarchy operations and by the actions of administrative roles.

4.4.1 Updates by administrative roles

Removing an element from admin-authority corresponds to removing an edge from the extended hierarchy. Therefore, (a, r) can be removed from admin-authority by role a' provided $a \in \mathcal{A}(a')$ and $r \in \mathcal{A}_S(a')$. If r were removed from $\mathcal{A}(a')$ as a result of deleting (a, r), then it is necessary to add (a', r) to admin-authority in order to preserve the administrative scope of a'. For example, given the admin-authority relation in Figure 4, DSO can remove ($\mathtt{PSO1}, \mathtt{PL1}$) from the relation. In this case it is not necessary to add ($\mathtt{DSO}, \mathtt{PL1}$) to admin-authority since ($\mathtt{DSO}, \mathtt{DIR}$) $\in$ admin-authority and hence $\mathtt{PL1} \in \mathcal{A}(\mathtt{DIR})$. Similarly, (a, r) can be added to admin-authority by role a' provided $a \in \mathcal{A}(a')$ and $r \in \mathcal{A}_S(a')$.

4.4.2 Updates by hierarchy operations

It may be necessary following a role hierarchy operation to update admin-authority in order to maintain administrative scope or to eliminate redundancy. Figure 5, based on the hierarchy and admin-authority relation in Figure 4, shows examples of such situations and provides a schematic motivation for the behaviour of the model. The operation in Figure 5b is performed by DSO; the other operations are performed by PSO1. We assume that edges implied by transitivity which would be lost as a result of a hierarchy operation are made explicit following the operation. An example of this is the addition of the edge ($\mathtt{ED}, \mathtt{PE1}$) in Figure 5d following the deletion of the edge ($\mathtt{ENG1}, \mathtt{PE1}$).

AddRole$(r, \Delta r, \emptyset)$ In this case r has no administrator(s). For example, in Figure 5a, we see that it is necessary to connect the new role X to the extended hierarchy. The obvious way to do this is to add ($\mathtt{PSO1}, \mathtt{X}$) to the admin-authority relation. Hence, the operation AddRole$(r, \Delta r, \emptyset)$ requires that (a, r) be added to the admin-authority relation.

DeleteRole(r) If $(a, r) \in$ admin-authority, then it is necessary to re-connect a to the extended hierarchy to preserve a's administrative scope. In this case we add (a, r') to admin-authority for all $r' \in \Delta r \cap \mathcal{A}(a)$.[4] For example, in Figure 5b, we add ($\mathtt{PSO1}, \mathtt{PE1}$) and ($\mathtt{PSO1}, \mathtt{QE1}$) to admin-authority.

AddEdge(c, p) If $(a, c) \in$ admin-authority, it may be possible that the addition of the edge (c, p) makes the tuple (a, c) redundant. Specifically, if $\uparrow c \setminus \uparrow (C(a) \setminus \{c\}) \subseteq \downarrow (C(a) \setminus \{c\})$ following the insertion of the edge, then we can remove (a, c) from admin-authority. For example, in Figure 5c, we remove ($\mathtt{PSO1}, \mathtt{PE1}$) from admin-authority.

[3] We use ranges because it is more economical than enumerating the elements in the role hierarchy.

[4] It may be that r' now occurs twice in admin-authority as a result of this procedure and hence several more deletions from admin-authority may be necessary because of the requirement that each role be controlled by a single role. Detailed algorithms for role hierarchy operations and their effect on the extended hierarchy are beyond the scope of this paper.

DeleteEdge(c, p) Finally, if $c \notin \mathcal{A}(a)$ following the deletion of the edge, then we add (a, c) to admin-authority. For example, in Figure 5d we insert (PSO1, ENG1).

5. A COMPARISON OF RRA97 AND RHA$_4$

Utility and applicability It is apparent from Proposition 3.2 and the conditions imposed on hierarchy operations that RHA$_4$ will be more "permissive" than RRA97, in the sense that it is less likely to cause hierarchy operations to fail. We confirm this observation by examining the sequence of hierarchy operations in Figure 1.

Given the role hierarchy in Figure 1a and the can-modify relation in Table 1, RRA97 does not permit operation AddRole(X, {QE1}, {DIR}) or AddRole(Y, $\emptyset$, {PE1}). This is essentially because the addition of the edge (QE1, X) or the edge (Y, PE1) would compromise the encapsulation of the range (ENG1, PL1). It does however permit operation AddEdge(PE1, QE1) because it occurs within an encapsulated range.

In RHA$_4$ it is possible for DSO to add X, although this causes the administrative scope of PSO1 to change (as shown in Figure 3). Similarly PSO1 can add Y and DSO can add the edge (PE1, QE1). It is immediately obvious that RHA$_4$ is a more permissive model than RRA97 because administrative scope can change dynamically. We do not believe that this is a disadvantage. (We reiterate that we do not believe that Y $<$ X is any more anomalous than PE1 $<$ QE1.)

We can immediately see that the requirement that an authority range be an encapsulated range imposes considerable limitations on the hierarchy operations that can be performed. This requirement also limits the number of hierarchies to which RRA97 can usefully be applied. For example, Figure 6a shows a hierarchy with no encapsulated ranges except (E, ED). Figure 6b shows the same hierarchy with a bottom element MinRole appended. This gives rise to a hierarchy with the same characteristics as a role graph [6]. However, it only introduces a single encapsulated range (MinRole, DIR), which does little to contribute to decentralized and autonomous administration of the hierarchy. In short, encapsulated ranges place strict requirements both on the nature of initial role hierarchies and on their subsequent development.

The hierarchy depicted in Figure 6a can easily be administered by RHA$_4$. In particular, the admin-authority relation defined in Figure 4a is perfectly suitable. In other words, RHA$_4$ is applicable to many more classes of role hierarchy than RRA97.

In RRA97 a new role can only have a single parent and child. That is, the RRA97 only supports the operation AddRole$(r, \{c\}, \{p\})$. (In particular, Y could not be added to the hierarchy in Figure 1a.) RHA$_4$ can support the creation of a new role with arbitrary numbers of parents and children.

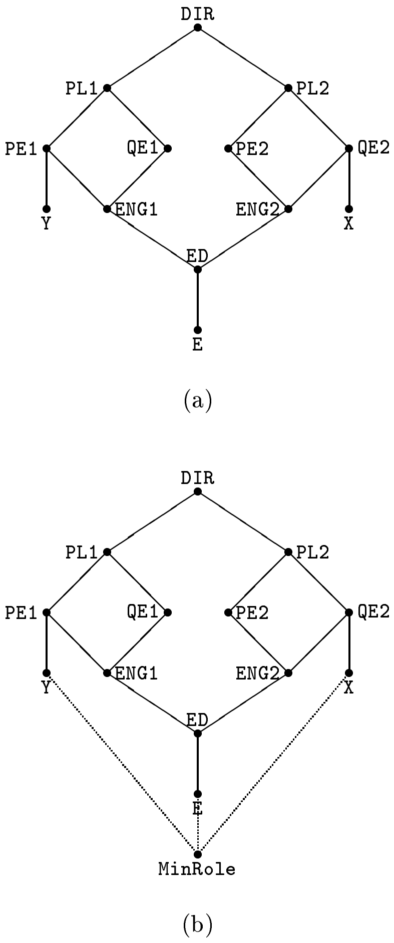
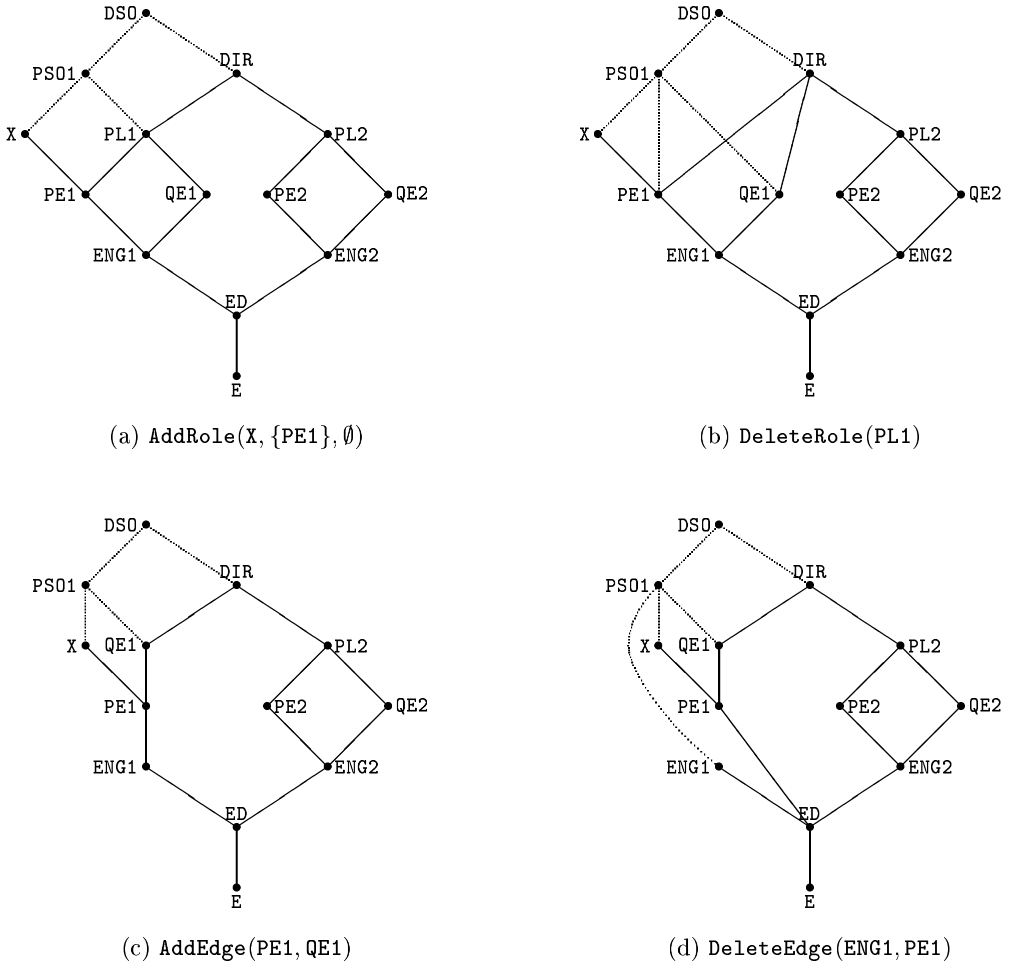


Figure 6: A problematic hierarchy for RRA97

Robustness RRA97 does not permit the deletion of a role r if it is the end point of any range in any ARBAC97 relation. (Of the seven ARBAC97 relations, five contain ranges.) The role r is "suspended" until such time as the required changes can be made to any relations affected by the deletion of r.

RHA$_4$ is not inconvenienced in this way by role deletions. Our forthcoming model for administration of user-role and permission-role assignments is defined using administrative scope not ranges, and hence special provision does not have to be made for the effect of role deletions.

Furthermore, ARBAC97 does not support updates to the can-modify relation in RRA97. That is, can-modify is, at worst, static and, at best, centrally administered. The admin-authority relation in RHA$_4$ is self-balancing in the sense that it is possible to define automatic update proce-



(a) AddRole(X, {PE1}, $\emptyset$)

(b) DeleteRole(PL1)

(c) AddEdge(PE1, QE1)

(d) DeleteEdge(ENG1, PE1)

Figure 5: Updates to the extended hierarchy

dures for it.

In short, the RHA$_4$ model, unlike RRA97, provides a coherent and self-contained framework for administration of the role hierarchy.

Comprehensibility and intuitive appeal It is clear that the admin-authority relation is far simpler than can-modify. In particular, the only constraints on tuples $(a, r) \in$ admin-authority are that $a \not< r$ and $(r, a) \notin$ admin-authority, while every authority range in can-modify must be an encapsulated range and any pair of authority ranges must not overlap. Furthermore, in the RHA$_4$ model we can visualize the administration of the role hierarchy using the extended hierarchy. In short, there is an intuitive and immediate interpretation of the admin-authority relation that is lacking in the can-modify relation.

A formal analysis of the complexity of implementing RHA$_4$ is beyond the scope of this paper. However, we note that the can-modify relation is defined in terms of encapsulated ranges which are subsets of R. There are $2^{|R|}$ subsets of R. Hence the number of tuples in can-modify is bounded by $|AR| \cdot 2^{|R|}$, where AR is the set of administrative roles.[5] The number of tuples in admin-authority is bounded by $|R|$. It can be seen from (5)–(7) that the complexity of an algorithm to determine whether a range is encapsulated is certainly no better than that of an algorithm to determine whether a role is in the administrative scope of another role. In other words, it is more attractive to implement RHA$_4$ than RRA97.

Integration The ARBAC97 model consists of three submodels URA97, PRA97 and RRA97. The URA97 model for user-role assignment and the PRA97 model for permission-role assignment were developed first, presumably because

[5] We note that this is a rather coarse upper bound as the number of encapsulated ranges will generally be significantly less than $2^{|R|}$.

of the relative simplicity of the task. These models exhibit some extremely useful behaviour, but the integration of the later RRA97 model with URA97 and PRA97 and the impact of hierarchy operations on URA97 and PRA97 relations is not well understood or defined. In particular, the effect of hierarchy operations on URA97 constraints and PRA97 constraints [7] is not considered.

We have developed a complete model for administration in a role-based access control context. The development of the model is based on the concept of administrative scope and hierarchy operations. It is far easier to extend RHA_4 to a model for administration than to define separate models and then try to integrate the models. Our model for administration requires three relations rather than the seven which appear in the ARBAC97 model. Furthermore, these relations are flexible and sensitive to hierarchy operations. That is, our model is self-contained and supports decentralized administration.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have defined the concept of administrative scope and used it to construct a series of models for administering a role hierarchy. This culminated in the RHA_4 model which provides a self-contained model for role hierarchy administration.

In the preceding section we discussed the relative merits of RHA_4 and RRA97, and found that our model is more attractive than RRA97 according to several different criteria. In short, we believe that our model offers significant practical and theoretical advantages over RRA97.

A couple of points worth noting about the extended hierarchy is that it can be used even when the set of roles is unordered (that is, there is no role hierarchy), as in OASIS [10], for example. It can also be used to administer a set of groups which naturally form a hierarchy under subset inclusion. That is, we can envisage a set of administrative subjects and an `admin-authority` relation where the most senior administrative subject assigns users to the largest groups and devolves the responsibility of assigning users to more specialized groups to less senior administrative subjects.

Our immediate priorities are to develop algorithms to implement RHA_4 (in the context of RBAC96) and to complete work on our administrative model based around RHA_4 in which we extend the use of administrative scope to user-role and permission-role assignment in a natural way. We are currently assessing the advantages of our model over ARBAC97 using similar criteria to those in Section 5. Early indications are that our model for administration performs significantly better than ARBAC97.

7. REFERENCES

[1] J. Crampton. *Antichains and authorizations.* PhD thesis, Birkbeck College, University of London, United Kingdom, 2002.

[2] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press, Cambridge, United Kingdom, 1990.

[3] S.I. Gavrila and J.F. Barkley. Formal specification for role based access control user/role and role/role relationship management. In *Proceedings of Third ACM Workshop on Role-Based Access Control*, pages 81–90, Fairfax, Virginia, 1998.

[4] V. Gligor. Characteristics of role-based access control. In *Proceedings of First ACM Workshop on Role-Based Access Control*, pages II9–II14, Gaithersburg, Maryland, 1995.

[5] J. Joshi, A. Ghafoor, W.G. Aref, and E.H. Spafford. Digital government security infrastructure design challenges. *IEEE Computer*, 34(2):66–72, 2001.

[6] M. Nyanchama and S. Osborn. The role graph model and conflict of interest. *ACM Transactions on Information and System Security*, 2(1):3–33, 1999.

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

[8] R.S. Sandhu, E.J. Coyne, H. Feinstein, and C.E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[9] R.S. Sandhu, D.F. Ferraiolo, and D.R. Kuhn. The NIST model for role-based access control: Towards a unified standard. In *Proceedings of Fifth ACM Workshop on Role-Based Access Control*, pages 47–63, Phoenix, Arizona, 2000. http://www.acm.org/sigsac/nist.pdf.

[10] W. Yao, J. Bacon, and K. Moody. A role-based access control model for supporting active security in OASIS. In *Proceedings of Sixth ACM Symposium on Access Control Models and Technologies*, pages 171–181, Chantilly, Virginia, 2001.