

UNDERSTANDING AND DEVELOPING ROLE-BASED
ADMINISTRATIVE MODELS

Jason Crampton

Information Security Group, Royal Holloway, University of London

12th ACM Conference on Computer and Communications Security

ADMINISTRATIVE MODELS

An *administrative model* is a formal representation of a higher level access control mechanism

- Determines whether a change to the state is permitted
- Determines how the state may evolve

Harrison, Ruzzo and Ullman described an administrative model for the protection matrix

- The state in the protection matrix model is determined by the set of subjects S , the set of objects O , and the matrix M
- Each of the primitive operations in the HRU model changes one of S , O or M

ANSI RBAC

The ANSI RBAC standard defines a number of basic components

- a set of users U
- a partially ordered set of roles (R, \leq)
- a set of permissions P
- a user-role assignment relation $UA \subseteq U \times R$
- a permission-role assignment relation $PA \subseteq P \times R$

A request by u to invoke permission p is granted if there exists $r, r' \in R$ such that $(u, r) \in UA$, $(p, r') \in PA$ and $r \geq r'$

MANAGING RBAC SYSTEMS

An administrative model for RBAC must consider changes to each of R , \leq , UA and PA

In this talk we consider the following operations

- $\text{addRole}(a, r, C, P)$ ($C, P \subseteq R$)
- $\text{deleteRole}(a, r)$ ($r \in R$)
- $\text{addEdge}(a, c, p)$ ($c, p \in R$)
- $\text{deleteEdge}(a, c, p)$ ($c, p \in R$)

What conditions should the parameters satisfy in order for the operation to succeed?

- ANSI RBAC describes the effect of administrative operations but not how to decide whether they should be permitted

EXISTING APPROACHES

Assign administrative permissions to roles

- Analogous to HRU approach for protection matrix
- May be difficult to reason about evolution of state

Use structure of role hierarchy to control changes

- ARBAC97¹ is rather complex and restrictive
 - Based around the idea of encapsulated ranges
- RHA² is more flexible
 - Based around the idea of administrative scope

¹Sandhu *et al*, TISSEC, 1999

²Crampton and Loizou, TISSEC, 2003

MOTIVATION

The goals of this work are

- to characterize the effects of hierarchy operations in terms of administrative domain preservation
- to better understand and improve the ARBAC97 and RHA models
- to provide a general framework for designing role-based administrative models

OUTLINE

- Administrative scope
- Preserving administrative domains
- Improving ARBAC97
- Conclusion

ADMINISTRATIVE SCOPE

Informally s belongs to the administrative scope of $r \in R$ if

- $s \leq r$
- every element greater than s is comparable to r

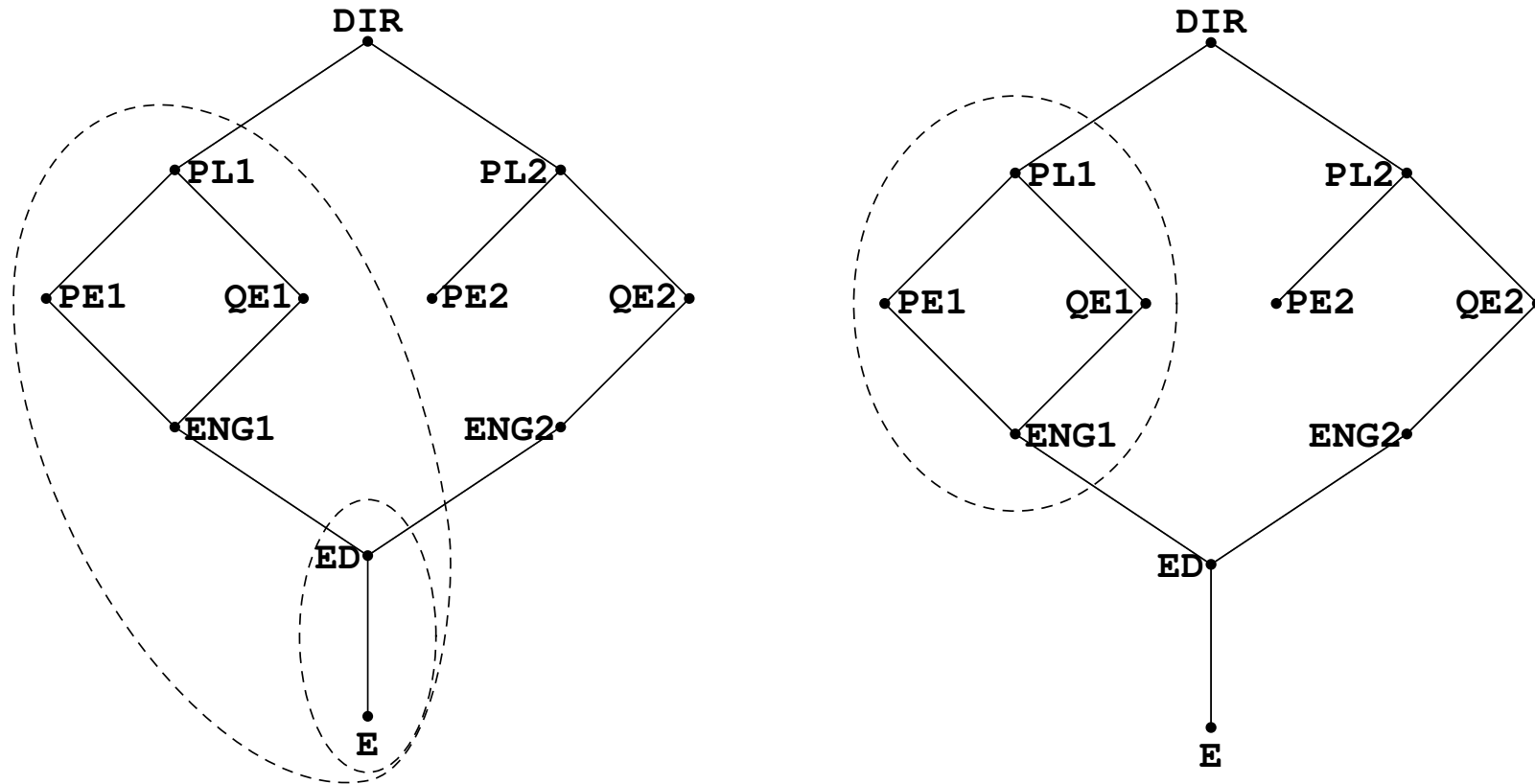
Define $\sigma : R \rightarrow 2^R$ where

$$\sigma(r) = \{s \in \downarrow r : \uparrow s \subseteq \downarrow r \cup \uparrow r\}^1$$

- The *administrative scope* of r is defined to be $\sigma(r)$
- The *strict administrative scope* of r is defined to be $\sigma(r) \setminus \{r\}$ and is denoted $\hat{\sigma}(r)$
- We will denote $\downarrow r \cup \uparrow r$ by $\updownarrow r$

¹ $\downarrow r = \{s \in R : s \leq r\}$; $\uparrow r = \{s \in R : s \geq r\}$

ADMINISTRATIVE SCOPE



ADMINISTRATIVE SCOPE AND HIERARCHY OPERATIONS

The RHA model allows role a to change those parts of (R, \leq) that belong to its administrative scope

Operation	Conditions
$\text{addRole}(a, r, C, P)$	$C \subseteq \hat{\sigma}(a), P \subseteq \sigma(a)$
$\text{deleteRole}(a, r)$	$r \in \hat{\sigma}(a)$
$\text{addEdge}(a, c, p)$	$c, p \in \sigma(a)$
$\text{deleteEdge}(a, c, p)$	$c, p \in \sigma(a)$

PRESERVING ADMINISTRATIVE SCOPE

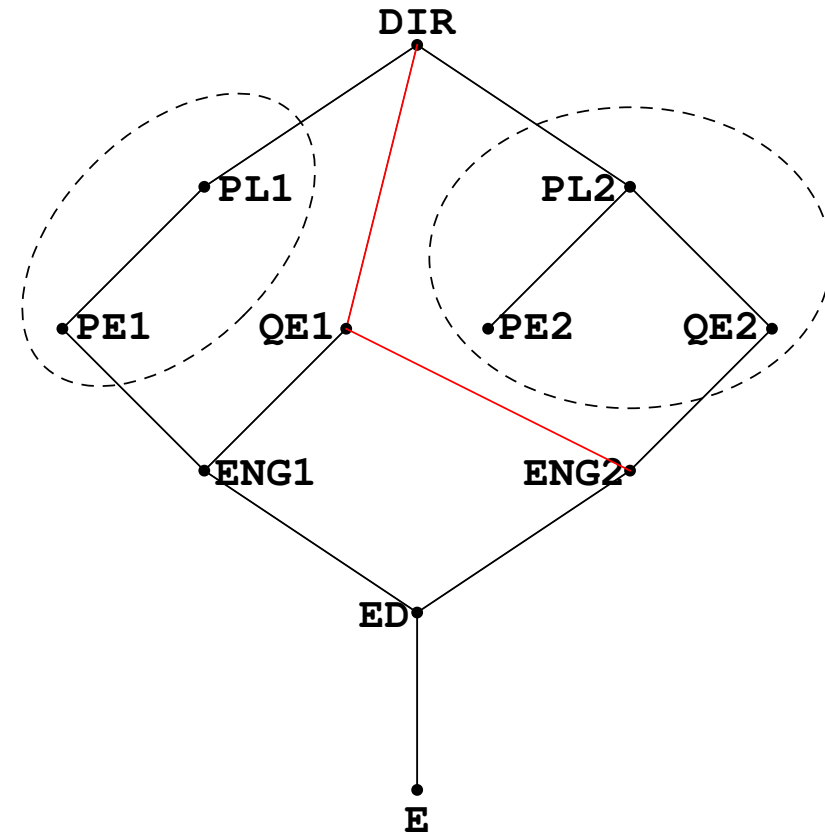
BREAKING ADMINISTRATIVE SCOPE

The RHA model allows r to break its own administrative scope

- `deleteEdge(PL1, QE1, PL1)`
creates an edge (QE1, DIR)

r can break the administrative scope of another role

- `addEdge(DIR, ENG2, QE1)`
breaks the administrative scope of PL2



PRESERVING ADMINISTRATIVE SCOPE

If r performs a hierarchy operation we could require that

- $\sigma(r)$ be preserved (0SP)
- $\sigma(r')$ be preserved for all $r' \geq r$ (1SP)
- $\sigma(r')$ be preserved for all $r' \in R$ (2SP)

THEORETICAL RESULTS

For convenience we will refer to $\sigma(r)$ as an *administrative domain*

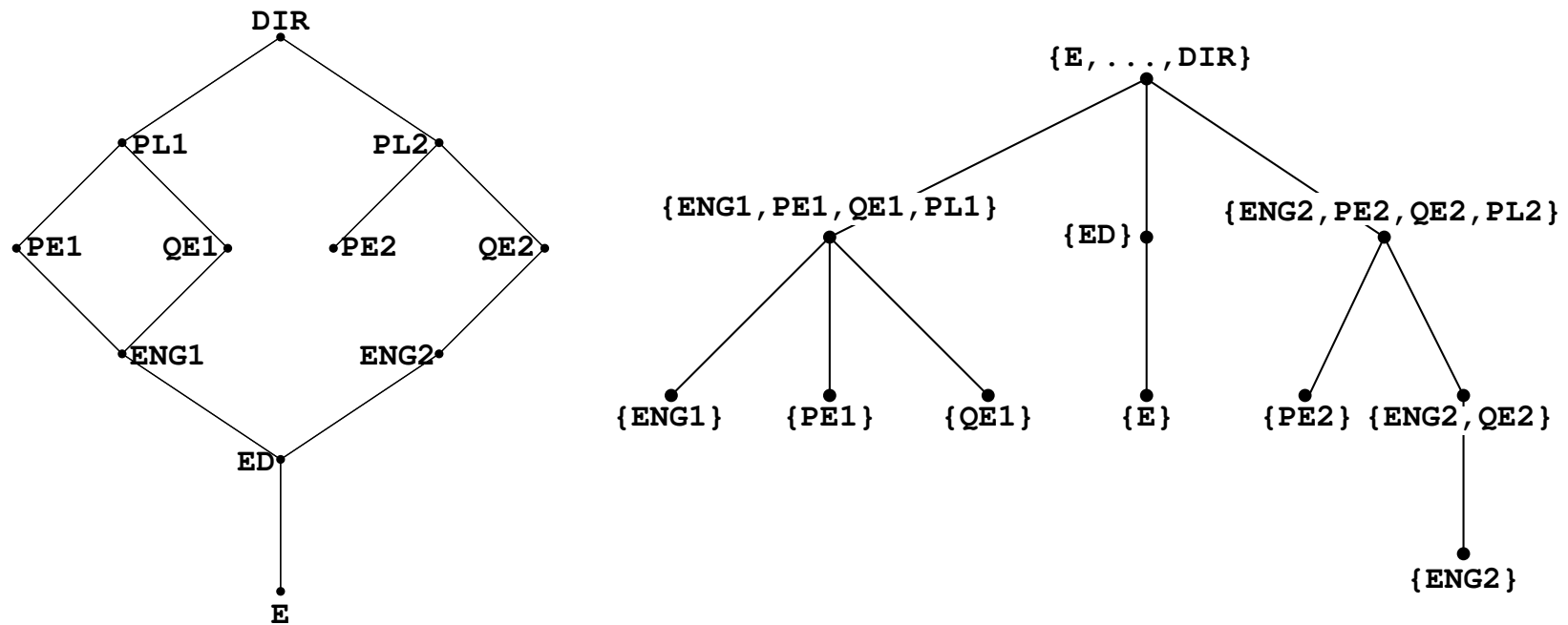
Lemma 1 *Administrative domains are either nested or disjoint*

Corollary 2 *The partially ordered set of administrative domains (\mathcal{D}, \subseteq) is a tree*

Corollary 3 *For all $r \in R$ there exists a smallest administrative domain containing r*

The smallest administrative domain containing r is called the *parent domain* and denoted $[r]$

THE ADMINISTRATIVE DOMAIN TREE



SOME NOTATION

$\lfloor X \rfloor$ is the largest domain D
such that $D \subseteq [x]$ for all $x \in X$

- $\lfloor \{QE1, ED\} \rfloor = \sigma(PL1)$

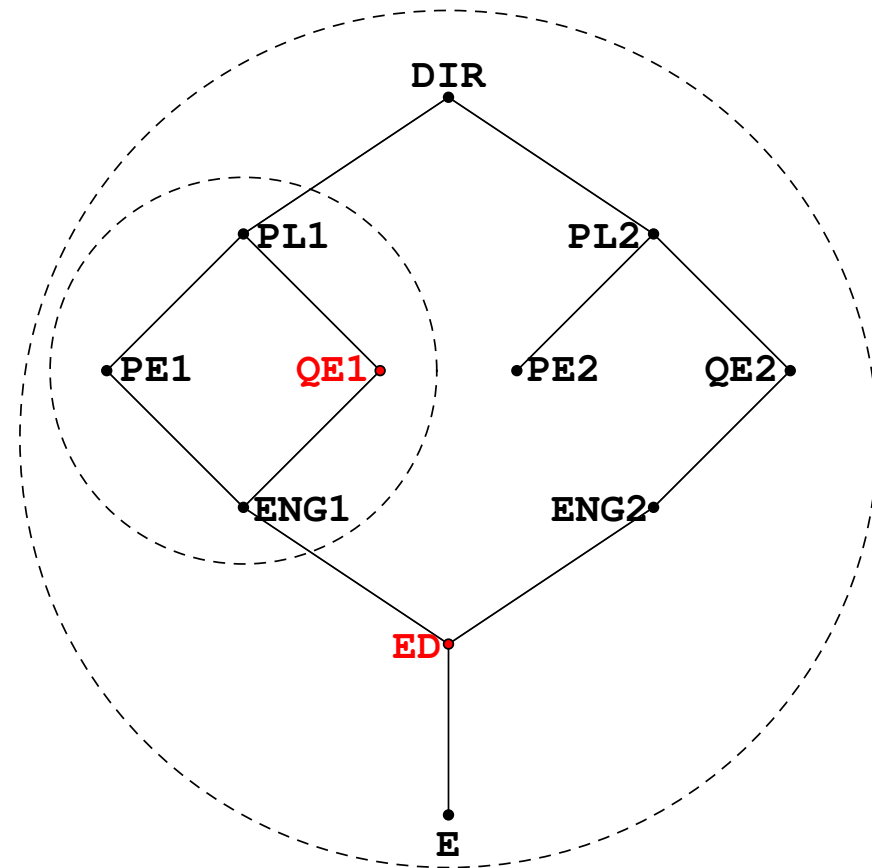
$\lceil X \rceil$ is the smallest domain D
such that $[x] \subseteq D$ for all $x \in X$

- $\lceil \{QE1, ED\} \rceil = \sigma(DIR)$

If $[x] = D$ for all $x \in X$ (and
some domain D) then

$$\lfloor X \rfloor = \lceil X \rceil = D$$

- $\lfloor \{PE1, QE1\} \rfloor = \lceil \{PE1, QE1\} \rceil$

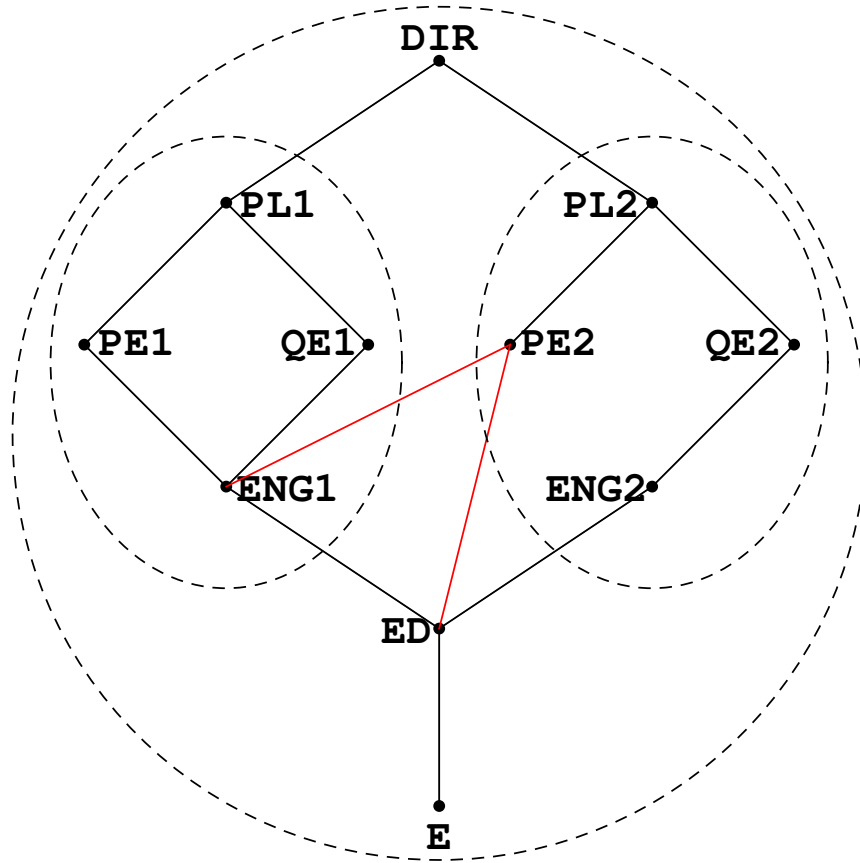


MORE RESULTS

Operation	Conditions		
	\mathcal{C}_{rha}	\mathcal{C}_0	\mathcal{C}_2
$\text{addRole}(a, r, C, P)$	$C \subseteq \hat{\sigma}(a)$ $P \subseteq \sigma(a)$	$C \subseteq \hat{\sigma}(a)$ $P \subseteq \sigma(a)$	$C \subseteq \hat{\sigma}(a)$ $P \subseteq \sigma(a)$ $[P] \subseteq [C]$
$\text{deleteRole}(a, r)$	$r \in \hat{\sigma}(a)$	$r \in \hat{\sigma}(a)$	$r \in \hat{\sigma}(a)$
$\text{addEdge}(a, c, p)$	$c, p \in \sigma(a)$	$c, p \in \sigma(a)$	$c, p \in \sigma(a)$ $[p] \subseteq [c]$
$\text{deleteEdge}(a, c, p)$	$c, p \in \sigma(a)$	$c, p \in \hat{\sigma}(a)$	$c, p \in \hat{\sigma}(a)$ $[\nabla p] \subseteq [c]$

 \mathcal{C}_0 is 0SP \mathcal{C}_0 is 1SP \mathcal{C}_2 is 2SP

EXAMPLE



- `addEdge(DIR, ED, PE2)` is 2SP (since $[PE2] \subseteq [ED]$)
- `addEdge(DIR, ENG1, PE2)` is not 2SP

PROOF METHOD

\mathcal{C}_0 is 0SP

- First prove that all RHA operations performed by a except `deleteEdge` preserve $\downarrow a$ and $\uparrow a$
- Proof is constructive and yields 0SP condition for `deleteEdge`

\mathcal{C}_0 is 1SP

- Follows from Lemma 1 and the fact that \mathcal{C}_0 is 0SP

\mathcal{C}_2 is 2SP

- \mathcal{C}_2 only allows edges to be added to the hierarchy if they are directed into interior domains
- Does not create any new senior roles for the child/children roles

IMPROVING ARBAC97

ARBAC97

Most well-known role-based administrative model

- Administrative counterpart of RBAC96 model

Administrative units are *encapsulated ranges*

- ARBAC97 requires encapsulated ranges to be nested or disjoint

Hierarchy operations are only permitted if

- arguments belong to an encapsulated range associated with the role performing the operation
- the operation preserves the encapsulation of every domain

It was not previously known how to check whether executing a hierarchy operation with a particular set of arguments would preserve all encapsulated domains

ENCAPSULATED RANGES AND ADMINISTRATIVE SCOPE

Definition 4 (Sandhu *et al*) A range $[x, y]$ is encapsulated if for all $z \in (x, y)$ and all $w \notin (x, y)$,

$$w > z \text{ iff } w \geq y$$

$$w < z \text{ iff } w \leq x$$

Lemma 5 For any encapsulated range $[x, y]$, $\llbracket [x, y] \rrbracket = \sigma(y)$

Corollary 6 Encapsulated ranges are either nested or disjoint

Proposition 7 The range $[b, t]$ is encapsulated iff for all $x \in [b, t]$,
 $\uparrow x \subseteq \uparrow t$ and $\downarrow x \subseteq \downarrow b$

ARBAC97 is a 2SP administrative model with encapsulated ranges (which are the symmetric analogue of administrative domains)

ARBAC97 REVISITED

Operation	Sandhu <i>et al</i>	Crampton	\mathcal{C}_2
$\text{addRole}(a, r, C, P)$	$C = \{c\}$ $P = \{p\}$ $c, p \in \hat{\sigma}(a)$ $[c] = [p]$	$C \subseteq \hat{\sigma}(a)$ $P \subseteq \sigma(a)$ $[C] = [P]^1$	$C \subseteq \hat{\sigma}(a)$ $P \subseteq \sigma(a)$ $[P] \subseteq [C]$
$\text{deleteRole}(a, r)$	$r \in \hat{\sigma}(a)$	$r \in \hat{\sigma}(a)$	$r \in \hat{\sigma}(a)$
$\text{addEdge}(a, c, p)$	$c, p \in \hat{\sigma}(a)$ $[c] = [p]$	$c, p \in \sigma(a)$ $[c] = [p]$	$c, p \in \sigma(a)$ $[p] \subseteq [c]$
$\text{deleteEdge}(a, c, p)$	$c, p \in \hat{\sigma}(a)$	$c, p \in \hat{\sigma}(a)$ $[c] = [p]$	$c, p \in \hat{\sigma}(a)$ $[\nabla p] \subseteq [c]$

¹ $[C] = [C] = [P] = [P]$

ARBAC97 AND RHA

The RHA model allows administrative domains to be broken

- Might be considered to be too permissive
- Widely applicable

The ARBAC97 model is applicable to few hierarchies

- Encapsulated ranges are rare
- No tree contains encapsulated ranges

Perhaps the best compromise is a 2SP model that uses administrative domains as the basic unit of administration

CONCLUDING REMARKS

WHAT ELSE IS IN THE PAPER?

Formal definition of what it means for a hierarchy operation to preserve $X \subseteq R$

Concept of preservation of autonomy (3SP)

- A hierarchy operation is only permitted if it is performed by the most local administrator
- Prevents “senior roles” from making changes to nested domains
- Any set of conditions that is 3SP is also 2SP

General framework for developing role-based administrative models

- RHA and ARBAC97 models are special cases

CONTRIBUTIONS

Characterization of role-based administrative models based on the extent to which hierarchy operations preserve administrative structure

- Concise and improved characterization of ARBAC97 and RHA models
- Development of general framework for role-based administrative models

The new characterization of RHA and ARBAC97 admits an efficient method for checking whether a hierarchy operation should be permitted

- Use the domain tree
- Such methods were not previously known

FUTURE WORK

Administration of UA and PA relations

- If a assigns u to a role $r \in \sigma(a)$ does that implicitly assign permissions outside $\sigma(a)$? If so, should the assignment be allowed?

Administrative separation of duty

- Administrative roles associated with administrative domains
- One (personnel?) role maintains the UA relation
- One (operational?) role maintains the PA relation
- One (managerial?) role controls R

FUTURE WORK

Development of API for role-based administration

- Implement RHA and ARBAC97 functionality
- Provide ability to “pick’n’mix” features from different administrative models

Administrative models for more complex RBAC models

- Multiple hierarchies

Safety problem for administrative models with different scope preserving properties

- We conjecture that the safety problem is likely to be easier to decide in a model that preserves domains to some degree

QUESTIONS