A Role-Based Delegation Model and some extensions

By:

Ezedin S.Barka

Ravi Sandhu

George Mason University

What is delegation?

- The basic idea behind delegation is that some active entity in a system delegates authority to another active entity to carry out some function on behalf of the former

Forms of delegation

- Delegation in computer systems can take many forms:
  - – human to machine
  - – machine to machine
  - – human to human
  - – perhaps even machine to human
- Our focus is on the Human to Human (where we consider the ability of a user who is a member of a role to delegate his role to another user who belong to another role).

RBAC96 is the base for our work

- We used the Role-Based Access Control Model, developed by Sandhu, as our framework

The RBAC96 Model

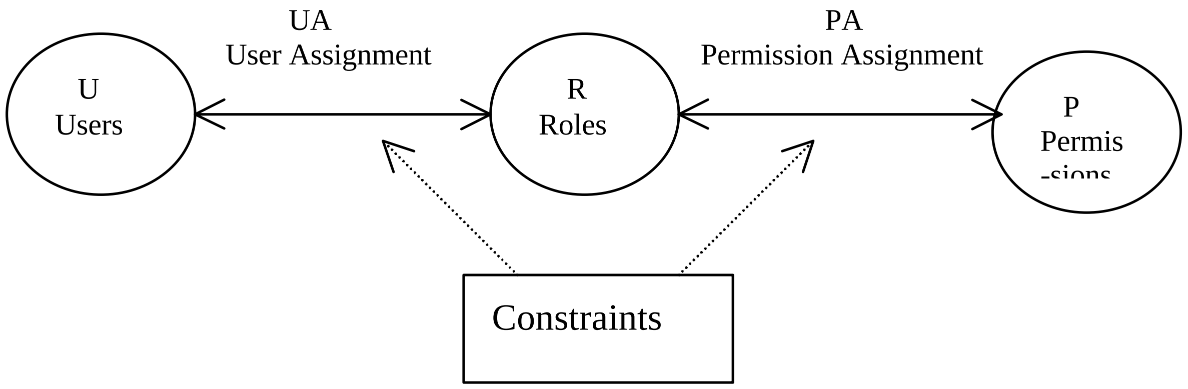


Figure 1-a: Simplified version of RBAC96 Model

Role-based delegation model-Flat roles (RBDM0)

- Assumptions &basic elements
  - Delegation between members in the same role is not allowed because it is meaningless.
  - delegation addressed in this model is a one step delegation
  - The delegation is total
  - Each delegating role r has two types of members, Original members Users_O(r), and Delegated members Users_D(r)

RBDM0

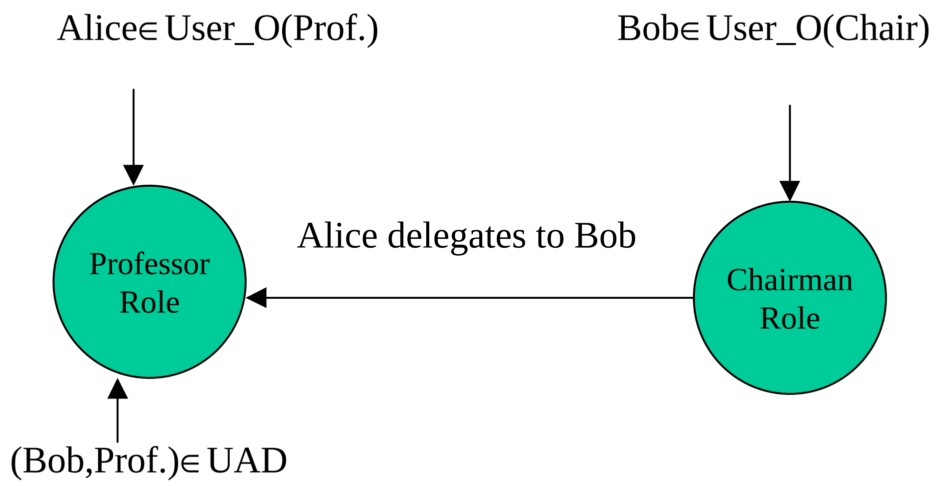
- Has the following components:
  - UAO $\subseteq$ U$\times$R many to may original member to role assignment relation
  - UAD $\subseteq$ U$\times$R , many to may delegated member to role assignment relation
  - UA = UAO $\cup$ UAD
  - UAO $\cap$ UAD =$\emptyset$ Original members and delegated members in the same role are disjoint

RBDM0..Cont.

- User_O(r) = $\{U \mid (U,r) \in UAO\}$
- User_D(r) = $\{U \mid (U,r) \in UAD\}$
- User_O(r) $\cup$ User_D(r) in a role get all the permissions assigned to that role
- Note that O(r) $\cap$ D(r) $= \emptyset$ because UAO $\cap$ UAD $= \emptyset$
- T is a set of duration
- Delegate roles: UAD$\rightarrow$T is a function mapping each delegation to a single duration

RBDM0..Cont.

- Role-to-role delegation is authorized by means of can-delegate relation: can delegate $\subseteq$ R×R. For example,



RBDM0..Cont..

- Revocation in RBDM0
  - Revocation using timeout
    - Simple & self triggering
    - Not enough, damage can happen within the duration
  - Grant dependent revocation
    - gives the power to the original members
    - No need to to define a can-revoke relation

Extensions

- We started by developing a very simple delegation model, RBDM-FR
- We are moving toward developing more complex models by evolving the simple models to include some extensions such as: Hierarchical roles,Muti-step delegation, …etc.

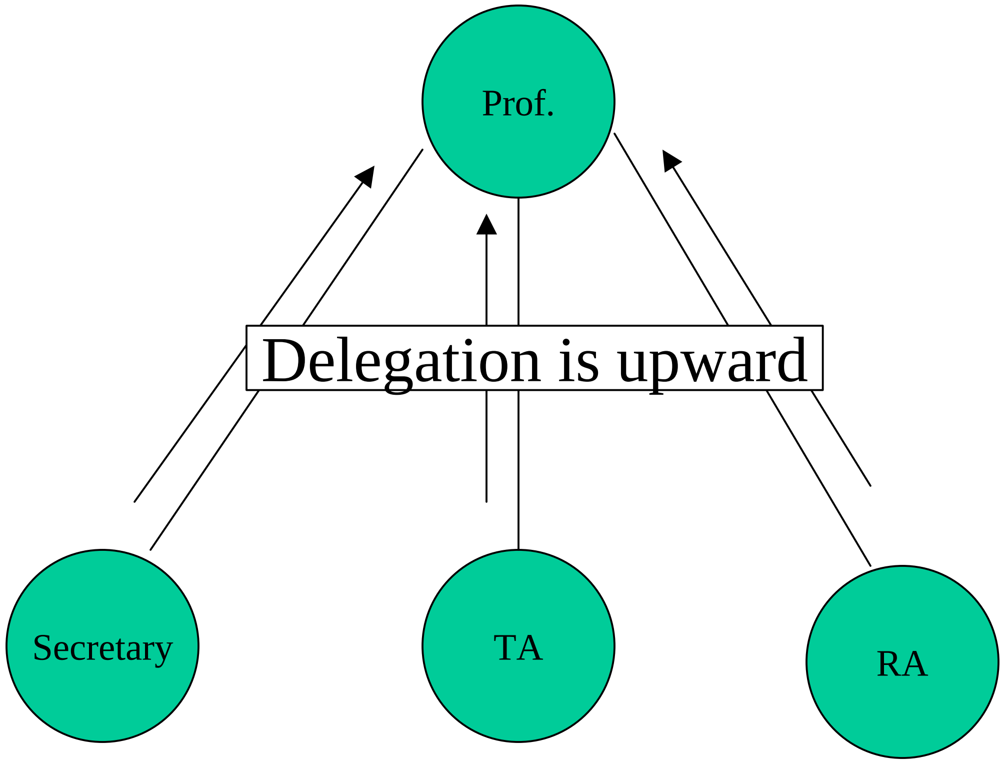
Extensions Cont..

- Extensions of RBDM0 include:
  - – Delegation in hierarchical roles
  - – Multi-step delegation
  - – There are two types of permissions
    - Delegable and Non-delegable permissions
  - – Grant-dependent revocation

Extensions Cont.

- Delegation in hierarchical roles
  - senior roles inherit the permissions of roles that are junior to them
  - adds more complications, because in hierarchical roles there are three possible ways for doing delegation
    - **Upward delegation**
    - **Downward delegation**
    - **Cross sectional delegation**

Example of delegation in hierarchical roles



RBDM-HR

- Has the following components:
  - RH $\subseteq R \times R$ is partially ordered role hierarchy (this can be written as $\geq$ in infix notation)
  - UAOE $\subseteq U \times R$ is many to many original explicit members to role assignment relation
  - UADE $\subseteq U \times R$ is many to many delegate explicit member to role assignment relation
  - UAO I $\subseteq U \times R$ is many to many original implicit member to role assignment

RBDM-HR..Cont..

– UADI $\subseteq$ U $\times$ R is many to many delegate implicit member to role assignment relation

– UA = UAOE $\cup$ UADE

– UAOE $\cap$ UADE = $\emptyset$ original explicit members and delegate explicit members in the same role are disjoint

– All members, Users_OE(r) $\cup$ Users_OI(r) $\cup$ Users_DE(r) $\cup$ Users_DI(r) in a role get all the permissions assigned to that role

RBDM-HR..Cont...

- Note that $(\forall r' \leq r)$ [User_OE(r) $\cap$ User_DE(r') = $\emptyset$] because UAOE $\cap$ UADE = $\emptyset$
- In RBDM-HR the semantics are defined such that there is a strict precedent among these two combinations as following:
- User_OE(r) > User_OI (r) > User_DE(r) >User_ DI (r)
- Delegate member: UADE $\cup$ UADI $\rightarrow$ T is a function mapping each explicit or explicit delegate membership in a role to a single duration

RBDM-HR..Cont...

- Role-to-role delegation is authorized by means of can-delegate relation:

can delegate $\subseteq R\times 2^{R}$

Multi-step delegation

- allows the delegated role memberships to be further delegated to other roles
- The RBDM0 will have the following components:
  - U, R, P are sets of users, roles , and permissions
  - $UA \subseteq U \times R$ is many to many user to role assignment relation
  - $UAO \subseteq U \times R$
  - $UAD \subseteq U \times R$
  - $UADD \subseteq U \times R$
  - $UA = UAO \cup UAD \cup UADD$
  - $UAO \cap (UAD \cup UADD) = \emptyset$
  - Users: $R \rightarrow 2^U$ is a function mapping each role r to a set of users

Multi-step delegation. Cont.

- The RBDM0 will have the following components:
  - Users(r) = $\{U \mid (U, r) \in UA\}$
  - Users_O(r) = $\{U \mid (U, r) \in UAO\}$
  - Users_D(r) = $\{U \mid (U, r) \in UAD\}$
  - Users_DD(r) = $\{U \mid (U, r) \in UADD\}$

Note that user_O(r) $\cap$ user_D(r) $\cap$ DD_(r) = $\emptyset$ because UAO $\cap$ UAD $\cap$ UADD = $\emptyset$

Types of Permissions (delegable and non-delegable)

- – Will not have any impact on the delegation or revocation, because the only relevant element to delegation and revocation is the human
- – It adds an extra control on what can and can not be delegated.

Grant-dependent revocation

- only the delegating member is allowed to revoke the role he delegated
  - Pros:
    - It makes the process of revocation more controllable
    - It eliminates conflict between the original members
  - Cons:
    - have to keep track of who the sponsoring role is in order to do revocation
    - If the sponsoring role gets revoked from the sponsoring user, then we have to deal with issue of what to do with its delegated roles and how

Summary

- Described the motivation, intuition and outline of a new simple and a non-trivial model for user to user delegation using roles called RBDM (role-based delegation model)
- Identified and discussed a list of some possible directions by which this model can be extended, this list including, delegation in hierarchical roles, multiple-step delegation, types of permissions,and grant-dependent revocation.