Framework for Role-Based Delegation Models

Ezedin Barka and Ravi Sandhu
Laboratory of Information Security Technology
Information and Software Engineering Department
George Mason University, Fairfax, VA 22030, USA
{ebarka, sandhu}@ isse.gmu.edu

Abstract

The basic idea behind delegation is that some active entity in a system delegates authority to another active entity to carry out some functions on behalf of the former. Delegation in computer systems can take many forms: human to human, human to machine, machine to machine, and perhaps even machine to human. In this paper we focuses on the human to human form of delegation using roles. As we will see there are many different ways in which role-based human-to-human delegation can occur. We develop a framework for identifying interesting cases that can be used for building role-based delegation models. This will be accomplished by identifying the characteristics related to delegation, using these characteristics to generate possible delegation cases, and using a systematic approach to reduce the large number of cases into few useful cases which can be used to build delegation models.

1. Introduction

Role-based access control (RBAC) has received considerable attention as a promising alternative to traditional discretionary and mandatory access control [FCK95, SCFY96, San97]. A role is a semantic construct forming the basis for access control policy. In RBAC, permissions are associated with roles, and users are made members of appropriate roles based on their responsibilities and qualifications, thereby acquiring the permissions of these roles. In RBAC, users can be easily reassigned from one role to another, and roles can be granted new permissions by means of new applications as systems come online, and permissions can be revoked with regard to roles as needed. This greatly simplifies the security management.

The basic idea behind delegation is that some active entity in a system delegates authority to another active entity to carry out some functions on behalf of the former. Delegation in computer can be human to human, human to machine, machine to machine, and perhaps even machine to human. Most delegation models in the literature address human to machine and machine-to-machine delegation [Glad97], [ABLP96], [GM90], [VAS91]. Models for propagation of access rights also relate to delegation indirectly (e.g. HRU, TAM, ATAM, SPM, and the Take Grant model) [HRU76], [San97], [Lamp71]. Our focus is on human to human delegation. Specifically, we consider the ability of a user who belongs to a certain role to delegate his role to another user who belongs to another role. For example, a professor in a university who is also a member in an advising committee role can delegate his membership in the advising committee role to another professor who belongs to another committee role. This delegation can take the form of being permanent of temporary delegation. Moreover, the same professor can delegate only part of his professor role (i.e. instructor) to his assistant. This delegation can be only temporary.

This type of delegation has not received much attention in the literature so far. In this paper we propose a framework for building good cases that can be used for developing role-based delegation models. This is the first systematic attempt toward addressing the problem of delegation between human using roles. We emphasize that the delegation itself occurs within the computer system even though it is human to human.

Our proposed framework begins by identifying a number of characteristics related to delegation between human, using these characteristics to create an exhaustive combination of possible delegation cases, and implement a systematic approach to reduce the large number of possibilities to few useful cases. These cases will be used for formalizing aspects of delegation based on the Role-based Access Control Model (RABC96) [SCFY96]. This work will involve the investigation and formalization of role-based delegation models using nine different delegation characteristics, as will be defined in section 3.1.

Before doing so we give a short background and a formal definition of Role-Based Access Control Model (RBAC96).

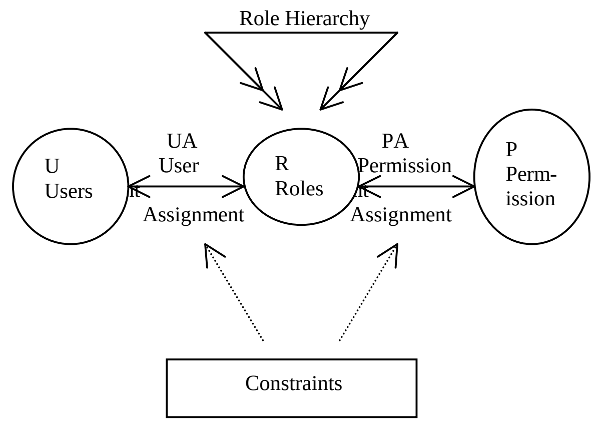

Figure 1: Simplified Version of RABC96 Model

2. The RBAC96 Model

The RBAC96 model, which was developed by Sandhu, *et al.* [SCFY96], is based on three sets of entities called Users (U), Roles (R), and Permissions (P) (see Figure 1).

A user (U) is a human being or an autonomous agent. A role (R) is a job title or a job function in the organization with associated semantics concerning responsibility and authority. A permission (P) is a description of the type of authorized interactions a subject can have with one or more objects.

Access control policy is embodied in RABC components such as user-role, role-permission, and role-role relationships. These RBAC components determine whether a particular user is allowed access to a specific piece of system data. A user can be assigned many roles, and a role can be assigned to many users. This property is captured by the many-to-many assignment relation called user assignment (UA). A role can be assigned many permissions, and a permission can be assigned to many roles. This property is captured by the many-to-many assignment relation called permission assignment (PA).

The formal definition for RBAC96 is as follows:

Definition 1: The RBAC96 has the following components:

1. U, R, P which are respectively the sets of users, roles, and permissions
2. UA $\subseteq$ U x R , which is a many-to-many user assignment relation assigning user to roles
3. PA $\subseteq$ P x R, which is a many-to-many permission assignment relation assigning permissions to roles.
4. RH $\subseteq$ P x R is a partial order on R called role hierarchy

3 RBDM Framework

Our approach to develop a framework for identifying interesting cases that can be used for building role-based delegation models between human begins with the identification of a number of characteristics related to delegation between humans. The identified characteristics include permanence, monotonicity, totality, administration, levels of delegation, multiple delegation, lateral agreements, cascading revocation, and grant-dependency revocation. We feel this list is a exhaustive list of characteristics although we would consider any additional characteristics that others may propose. Trying to address every characteristic as mutually exclusive is a formidable task, and it can be very complicated. Therefore, a systematic approach is used in order to reduce the large number of possible cases. These reduced cases, which can be useful in business today, can be used to build delegation models. The following section will provide definitions and explanations for these characteristics.

3.1 Definitions of characteristics:

The following is a list of definitions of the characteristics that are related to delegation:

1. Permanence

Permanence in role-based delegation models refers to types of delegation in terms of their time duration. Permanent delegation refers to delegation wherein another user who is a member of another role permanently replaces the delegating user. In this type of delegation, once the delegating user delegates his role, he or she can no longer take it back (but can get it back through the security office). Upon delegation, the user who received the delegation assumes the same full power as any of the original member of that role. For example, a professor who is a member of a role called advising committee, and is no longer able to serve in the committee, can permanently, delegate his membership of that role to some other professor. In this case the delegate professor becomes a permanent replacement to the delegating professor. Also, the delegating professor looses all of his permissions in that role, and can not get them back unless the security officer assign him back to that role. On the other hand, temporary delegation refers to the type of delegation that is limited by time. One that time is expired, the delegation is no longer valid. For example, that same delegating professor may elect to delegate his role only temporarily to his secretary to do some task on his behalf. In this case the delegating professor will not loose his power in that role, and can revoke that delegation as he wishes.

2. Monotonicity (monotonic/non-monotonic)

Monotonicity refers to the state of the power that the delegating role member possesses after he or she delegates the role. A monotonic delegation means that upon delegation the delegating role member maintains the power of his or her role. This user continues to be able to perform the same operations he or she had before delegating his or her role (e.g., he or she can revoke the membership of the delegated member and can re-delegate it to someone else).

On the other hand, with a non-monotonic delegation, upon delegation the delegating role member loses the power of the delegated role for the duration of the delegation. In this case, the delegating role member will no longer be able to use the permissions of the role he or she delegated. However, the delegating role member never loses the power of revoking the roles that he or she delegates. Therefore, once the delegation is revoked or expired, the delegating role member will regain full power over that role. For example, using the same professor analogy from the example above, if the delegating professor non-monotonically delegates his role to his secretary, then as long as that delegation is active the professor will not be to act in the delegated role. However, he is still responsible for the behavior of the delegate member in that role, which is his secretary in this case.

3. Totality

This term refers to how completely the permissions assigned to that role are delegated. There are two options: total delegation and partial delegation. Total delegation means delegating all of the permissions that are assigned to the delegated role. On the other hand, partial delegation means that only subsets of the delegated role are delegated. Partial delegation is much easier to address using role hierarchies. For example, a professor in a university who has teaching assistant, lab assistants, and a secretary may delegate only subset of his roles to different people. (E.g., he may delegate his teaching role to his teaching assistant, his research role to his lab assistant, and his administration role to his secretary in order to handle his e-mail).

4. Administration

The term administration will be used here to describe the actual administrator of the delegation. There are two administration types for delegation: the self-acted delegation, wherein the delegating role member him- or herself administers the delegation; and the agent-acted delegation, wherein the delegating role member nominates a third party (an agent) to conduct the delegation on his or her behalf. Regarding the latter, there are situations in which it is more convenient to have a third party administering the delegation on behalf of the user (e.g., in the case in which the delegator is not available, or is unable to perform the delegation). Usually, the agent can delegate to anyone else, but cannot delegate to him or herself. For example, a professor may designate a third party (e.g. another professor who is a member of the same professor role) to administer any of the delegations on his behalf.

5. Levels of delegation

This characteristic defines whether or not each delegation can be further delegated and for how many times. Single step delegation does not allow the delegation to be further delegated; this means that the delegated member is not allowed to further delegate the role that is been delegated to him or her. Two- or multi-step delegation allows the delegated member to delegate further his or her delegated role to a third user, and so on. For example, in our example above, once a professor delegates his role to another professor, the second may further delegates that role to someone else. Changing the policy to a single step delegation, which can enforce the delegation to take place only once can stop this.

6. Multiple delegation

This type of delegation refers to the number of people to whom a delegating role member can delegate at any given time. Sometimes, a delegating role member needs to delegate his or her role to more than one person at the same time. For example, a professor may need to delegate his or her role to more then one of his or her research assistants. This type of delegation is more effective if the delegation is temporary, because, delegating a role (permanently) to more than one person at the same time can create some accountability problems.

7. Agreements

The term agreement is used here to address the delegation protocol between the delegator and the delegated members. It is of two types: bilateral agreement and unilateral agreement. A bilateral agreement is an agreement wherein delegation is accepted by both the delegating role member and the delegated member. The delegating member is agreeing to delegate the role, and the delegated member is agreeing to accept the role. For example, before a professor delegates his role to another professor (or to any of his assistants), the second has to first agree to accept that responsibility before the delegation can take place. A unilateral agreement, on the other hand, is a one-way decision. Only the delegating role member can decide to delegate the role. Once the delegating member identifies a role member to whom he or she delegates the role, the delegated member has to accept that responsibility. The security implication for this type of delegation is that people sometimes use it for some denial of service. In systems where every user is assigned a certain quota of memory, someone can create some files and delegate them to someone else causing the second person to exceed his limit, which can lead to a denial of service.

8. Revocation

Revocation refers to the process by which a delegating user can take away the privileges that he or she delegated to another user r who is a member of another role. There are some interesting issues involving revocation: among these issues are those raised by cascading revocation and by grant-dependency revocation. The following is a brief description of these types of revocations:

8.1 Cascading revocation

This type of revocation refers to the indirect revocation of membership as a result of the revocation of the membership of some other related roles (i.e., supporting role or sponsoring role). With the supporting role (the role to which the delegated user belonged prior to being a delegated member of the new role), if the delegated member loses his or her membership in his or her supporting role, then as a result he or she will lose his or her delegated membership in the new role. In the case of the sponsoring role (the role that is responsible for bringing in the delegated member), if that role is revoked, then the membership that was delegated by that revoked role will also be revoked.
For example: suppose that Alice who is an original member of role a, Bob is an original member of role b. Now suppose that Alice delegates her role to Bob who delegate that same

role to another user, say Charlie. Then, revocation will have the following scenarios:

- If Alice revokes Bob's delegate membership in a, then as a result Charlie will also looses his delegate membership in a.
- If Alice looses her membership in a then Both Bob and Charlie will loose their delegate memberships in a.
- If Bob looses his original membership in his own role (b) then he will also loose his delegate membership in a, and consequently, Charlie will also looses his membership in a.

8.2 Grant-dependency (grant-dependent/grant-independent)

Grant-dependency refers to the decision that has been made regarding who may revoke the membership of the delegated member in a role. In the case of grant-dependent delegation, only the delegator is allowed to revoke the membership of the delegated member. Grant-independent delegation, on the other hand, allows any member in the sponsoring role to revoke the membership of the delegated member. This decision becomes important, for example, if the delegated member behaves badly: with grant-dependent delegation, it could take a long time (depending on the delegator) to revoke the offender's membership; with grant-independent delegation, that membership can be revoked quickly by any original member of that role (which may of course create some friction between the original members).

Using different combinations of the above characteristics will give us a very large number of possible modes in which to do delegation. Therefore, we have implemented a systematic approach in order to reduce this large number of cases into some useful ones.

The following section will explain the framework we used to identify the useful cases for developing role-based delegation modes.

3.2 Reduction approach

We have identified a systematic approach by which we can reduce the large number of possible cases into a few useful ones.

We first partitioned delegation based on its permanence (permanent or temporary delegation). We believe it is useful to develop delegation models that support the implementations of the permanent and temporary delegation policies. We further partitioned both the permanent and temporary delegations. The permanent delegation was partitioned based on its monotonicity, whereas the temporary delegation was partitioned based on its level of delegation. Finally, we partitioned single step delegation of the temporary delegation based on its monotonicity. After the partitioning was done, we added the rest of the characteristics (one at a time) to each node and tested for combinations that are useful in business today and that can be used in developing delegation models. Figure 2 shows the combinations that make sense.

On the permanent side, we could make the claim that there is one clear path that can be followed to develop a delegation model. That path includes the following characteristics: permanent, non-monotonic, self-acted, and total delegation. Other characteristics do not have much effect on the model, and were ignored.

On the temporary side, however, there are a number of possibilities; therefore, we have to make some simplification in order to identify useful combinations and ultimately to develop one comprehensive model for formulating the user-to-user delegation. The simplification is to eliminate the multi-step delegation, given that dealing with multi-step delegation is a very complicated issue.

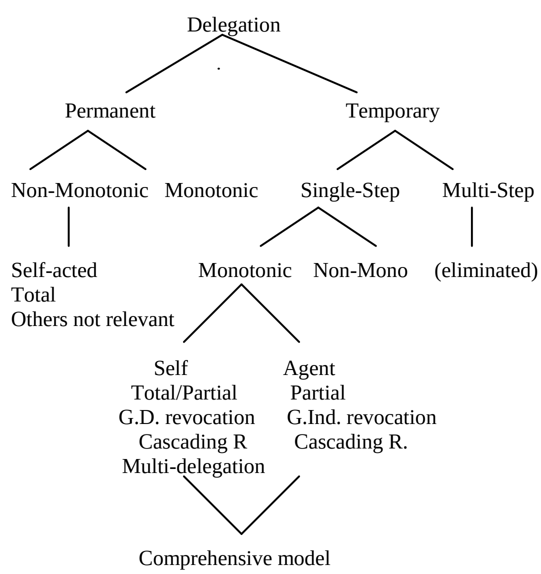

Figure2: Tree structure showing the partitioning process

The following sections explain the testing process and provide justification for selecting these combinations of characteristics. We start by examining the permanent side followed by the temporary side.

3.2.1. Permanent delegation

This section examines the process of identifying the one combination of characteristics useful for defining a role-based delegation model.

- Permanent and monotonicity

Permanent-monotonic delegation does not appear to be very desirable. If a user in a role elected to delegate his or her role, permanently, to another user who is not a member of that role, the delegated member becomes his or her replacement in that role. In such case there is no need for the delegating role member to maintain his or her power in that role

Permanent and non-monotonic delegation, on the other hand, seems to be more useful. Once a delegating user permanently delegates his or her role to another user and loses his or her power in that role, then he or she is no longer responsible for the behavior of the new delegated member. Non-monotonic delegation allows the new role member to act independently and as if he or she is an original member of that role. For example, a member of an advising committee (say, Alice) decides that she can no longer serve in the committee and is willing to permanently delegate her role to someone else who is not a member of this committee (say, Bob). In this scenario, it makes a lot of sense for the delegation by Alice to Bob to be non-monotonic. Alice loses all of her power in that role. In fact, to preserve the security of committee role, the delegation must be non-monotonic. This is because Alice is no longer a member of that role.

- Permanent, non-monotonic, and totality

When the delegation is permanent and non-monotonic, partial delegation is not desirable, because delegating only a subset of the role does not allow the delegated member to carry out the full task of the delegating member. For example, suppose that Alice (who is an original member of Role A), permanently-monotonically delegates only a subset of her role to Bob (who is a member of Role B). In this case Bob will not be able to carry out fully the task of Alice (e.g., if the task of reviewing student records was not delegated, then Bob will not be able to provide efficient advice to the students). Moreover, neither Alice nor Bob will end up with full power in that role.

- Permanent, non-monotonic, total, and administration

With permanent, non-monotonic, total delegation, having an agent doing the delegation does not seem to be very appealing; we think it is more sensible if the delegating member him or herself administers the delegation. Delegation of this kind is usually planned in advance, and there is no extreme need to rush it. Thus, there is no need for the delegating member to nominate someone else to do the delegation on his or her behalf

- Permanent, non-monotonic, total, self-acting, and levels of delegation

If the delegation is permanent and non-monotonic, then it is up to the delegated member to decide whether or not he or she wants to delegate further his or her role. Therefore, from a modeling perspective, the level of delegation is irrelevant.

- Permanent, non-monotonic, total, self-acting, and agreement

With this combination of characteristics, lateral agreement between the delegating member and the delegated member is essential, because if the delegated member does not accept the responsibility of the delegating role, then the purpose of the delegation is defeated and there is no need for it to take place.

- Permanent, non-monotonic, total, self-acting, and revocation

With permanent and non-monotonic delegation, revocation becomes irrelevant. Once a user becomes a permanent delegated member in a different role, then he or she assumes the full power in that role, and only the security officer can revoke his or her membership in that role.

In summary, with permanent delegation, it is desirable to let the model be simple, yet tight (no agent should be involved, and no partial delegation). Therefore, it makes more sense to choose the following combination: permanent, non-monotonic, self-acted, total delegation. Other characteristics are not relevant and therefore were ignored.

3.2.2 Temporary delegation

With temporary delegation, identifying specific paths to develop role-based delegation models is not as obvious as it was with permanent delegation. In the case of temporary delegation, there are lots of possibilities, and the process of selecting paths that make more sense for developing delegation models is quite complicated. For example, if we test for useful combinations using the administration characteristic in combination with the temporary delegation, we cannot make a clear determination as to which characteristic makes more sense (i.e., with the temporary delegation we cannot determine whether the self-acted or the agent-acted delegation makes more sense). Therefore, in the case of the temporary delegation:

- Everything is bilateral
- Only single step delegation is addressed
- Only monotonic delegation is addressed

The following section discusses the testing process related to temporary delegation. For simplicity, we used the administration characteristic to partition further these characteristics.

3.2. 2 .1 Self-acted delegation

- Temporary, self-acted, and monotonicity

In combination with temporary and self-acted delegation, monotonic delegation has two advantages. First, it prevents the delegated member from having exclusive power over the role, especially because the delegator is still responsible for that role. Monotonic delegation allows the delegator to review and correct any wrong action that might be taken by the delegated member. Second, the delegator can take over the task of the delegated member in case the delegated member suddenly loses his or her membership.

- Temporary, self-acted, monotonic, and totality

With these characteristics is seems useful to consider both the partial and the total delegation. The delegator should be able to delegate all of his or her role or just part of it. In both cases, this will not conflict with the permanent, self-acted, monotonic delegation.

- Temporary, self-acted, monotonic, totality, and levels of delegation

Most published research in the area of the Discretionary Access Control (DAC) suggests that multi-step delegation is a very complicated issue and that dealing with it can create a lot of problems; therefore, with temporary delegation, multi-step delegation is eliminated from our consideration. We will consider only single step delegation. The multi-step delegation issues will be deferred for now, and multi-step delegation will be discussed later on with one of the special case models that we are going to develop.

- Temporary, self-acted, monotonic, totality, single-delegation, and multiple delegation

With the delegation being Temporary, there are situations where is it more useful to have the delegator him self delegate his role to deferent other users, (i.e., where he only trusts specific individuals to do a certain task). Therefore, it is useful to consider multiple delegation in this case.

- Revocation

- With this combination of characteristics, it is more appealing to consider both types of revocation: grant-dependent and grant-independent revocation.

- It is also more appealing to consider the cascading revocation issues with this combination of delegation characteristics.

Therefore, with the delegator as the administrator of the temporary delegation, we select the path, temporary, self- acted, single step, monotonic, partial/total, delegation, grant-dependent/grant-independent, and cascading revocation

4. Conclusion

In this research, we have identified some characteristics related to delegation between human in computer systems. We then used a systematic approach to reduce the large number of possible cases to a few cases that can be useful in business today. We partitioned the characteristics into permanent and temporary delegations. In the permanent side it was very obvious to determine a path that can lead to the development of a delegation model. One the temporary side, there were a lot of possibilities which led to making more simplifications. These simplifications included the elimination of the multi-step, and the non-monotonic delegations. Consecuently, on the temporary side, it is more feasible to develop a comprehensive model that can capture all, or most, of the characteristics related to temporary delegation

References

[ABLP96] Martin Abadi, Michael Burrows, Butler Lampson and Gordon Plotkin. A calculus for Access Control in Distributed Systems. ACM Transactions on Programming Languages and Systems, Vol. 15, No 4, September 1993, pages 706-734.

[FCK95] David Ferriaolo, Janet Cugini, and Richard Kuhn. Role-based access control (RBAC): Features and motivations. In Proceedings of 11th Annual Computer Security Application Conference, pages 241-48, New Orleans, LA, December 11-15 1995.

[FK92] David Ferriaolo and Richard Kuhn. Role-based access controls. In Proceedings of 15th NIST-NCSC National Computer Security Conference, pages 554-563, Baltimore, MD, October 13-16 1992.

[Glad197] Henry M. Gladny, Access Control for Large Collections. ACM Transactions on Information Systems, Vol.15, No.2, April 1997, Pages 154-194.

[GM90] Morrie Gasser, Ellen McDermott. An Architecture for practical Delegation in a Distributed System. 1990 IEEE Computer Society Symposium on Research in Security and Privacy. Oakland, CA. May 7-9, 1990.

[HRU76] M.H. Harrison, W.L. Ruzzo, and J.D. Ullman, Protection in Operating Systems. Communications of ACM. 1976. Pages 461-471.

[Lamp71] B.W. Lampson, Protection. 5th Princeton Symposium on information science and systems. Pages 437-443.

[San92] Ravi Sandhu, The Typed Access Matrix Model. Proceeding Symposium on Security and Privacy, Oakland, CA, May 4-6, 1992, pages 122-136.

[San97] Ravi Sandhu. Rationale for the RBAC96 family of access control models. In Proceedings of the 1st ACM Workshop on Role-Based Access Control. ACM, 1997.

[SB97] Ravi Sandhu and Venkata Bhamidipati. Role-based administration of user-role assignment: The UR97 model and its Oracle implementation. In Proceedings of IFIP WG11.3 Workshop on Data Security. August, 1997.

[SCFY96] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. IEEE Computer, 29(2):38-47, February 1996.

[VAS91] Vijay Varadharajan, Philip Allen, Stewart Black. An Analysis of the Proxy Problem in Distributed systems. IEEE Symposium on Research in Security and Privacy. Oakland, CA 1991.