ACCESS CONTROL MODEL FOR GRID VIRTUAL ORGANIZATIONS

Nasser B., Benzekri A., Laborde R., Grasset F., Barrère F.
Insitut de Recherche en Informatique de Toulouse-IRIT, Paul Sabatier Univeristy, 118 route de Narbonne,Toulouse,France

Keywords: Grid, Distributed systems, Security, Access control, RBAC, OrBAC

Abstract: The problems encountered in the scientific, industrial and engineering fields entail sophisticated processes across widely distributed communities. The Grid emerged as a platform that has a goal enabling coordinated resources sharing and problem resolving in dynamic multi-institutional Virtual Organizations (VO). Though the multi-institutional aspect is considered in the grid definition, there is no recipe that indicates how to fabricate a VO in such environment where mutual distrust is a constraint. Excluding a central management authority, the different partners should cooperate to put in place a multi-administrated environment. The role of each partner in the VO should be clear and unambiguous (permissions, interdictions, users and resources to manage…). Organizing a large scale environment is error prone where not well formalized models lead to unexpected security breaches. Within the access control models RBAC has proved to be flexible but is not adapted to model the multi-institutional aspect. In this context, we propose a formal access control model, OrBAC (Organization Based Access Control model), that encompass concepts required to express a security policy in complex distributed organizations. Its generality and formal foundation makes this model the best candidate to serve as a common framework for setting up Virtual Organizations.

1 INTRODUCTION

The problems encountered in the scientific, industrial and engineering fields entail a sophisticated process across widely distributed communities involving huge data amounts to be managed, analyzed using complex computing applications and finally stored (Fedak,2001)(Foster, 1997)(Baru,1998). The needed cooperation across the communities is no more trivial file exchange but rather direct access to computers, software, data, and other resources. It involves hundreds of processes that necessitate acquiring resources dynamically and communicate efficiently.

The grid has emerged as a platform that has as goal enabling coordinated resource sharing and problem solving in dynamic multi-institutional virtual organization (Foster, 2001). The grid as a middleware leverages the cooperation level between the different partners higher than primarily connectivity level. It offers layers and protocols (see http://www.gridforum.org/) that hide the underlying infrastructure (computer, hard disk, CPU, services…), facilitate their exploitation, and employ them in different scenarios as Data grids (Baru,1998), Computing grids (Foster,1997) and Desktop grids (Fedak,2001). From conceptual point of view, the different communities in a grid environment, federating into an alliance for a certain goal, put in place a Virtual Organization. The Virtual Organization (VO) constitutes the common shared space between the different institutions. It embraces topics as shared resources (data, computer hardware and software), sharing relationships, fine-grained access control…

Though the Grid defines the VO notion, it doesn't indicate how to create and manage this large scale, multi-organizational structure. Constructing a VO necessitates tracing its boundaries, specifying the different access rights within it and assuring management during its life time.

Solutions as the Community Authorizations Service (CAS) (Cannon, 2003) allow enforcing access control policies within the VO using X.509 extensions. The extensions include low level user's permission to grid resources as read/write file permissions. Another system is the Virtual Organization Management System (VOMS) (Alfieri,2003), where a server supplies the user with

B. N., A. B., R. L., F. G. and F. B. (2005).
ACCESS CONTROL MODEL FOR GRID VIRTUAL ORGANIZATIONS.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 152-158
DOI: 10.5220/0002535001520158

an X.509 certificate extended with role and group information. The destination resource needs to know the access policies concerning the role/group to make authorization decisions.

Other projects as GRASP (Djordjevic, 2004) and DAME (Russell,2004) offer architectures based on X.509 certificates to enable formation of VOs responding to their proper needs.

These aforementioned solutions propose mechanisms to put in place a Virtual Organization by enforcing a certain access control policy. However there is no clear method how such policy would be conceived. Recalling that each partner needs to keep control on his assets where delegating others to specify the way to use his resources or its access control policy is often not acceptable.

A central management of the shared space is then excluded and the different partners should cooperate to create a scalable, dynamic, multi-administrated system.

As a first step, we think that the VO should refer to abstract entities rather than concrete users and resources. Abstraction improves scalability, enables adding users and resources dynamically according to the local administrative domain policy. As a second step, the relations between the different entities have to be specified. Informally specifying these relations is error prone causing potential security breaches. It is judicious to achieve these two steps within a formal access control model that governs the shared space (Samarati).

Towards a dynamic creation of virtual organizations in a multi-domain grid environment, we start by introducing the Grid along with the complexities confronted in a decentralized dynamic environment. The need for an access control model is emphasized and within this context RBAC and its limits are reviewed (Sandhu 1996)(Sandhu 1999). We then propose OrBAC (Organization based Access Control model) (Abou El Kalam, 2003) (Cuppens,2003) as the access control model that specifies the different relationships within the VO. OrBAC abstracts “users, object and action” with “role, view and activity” and models flexibly the different responsibilities within a VO. Finally, we show how the grid environment can be modelled using OrBAC and conclude with the future works.

2 GRID

The grid has emerged as a platform that has as goal enabling coordinated resource sharing and problem solving in dynamic multi-domain virtual organization (Foster 2001).

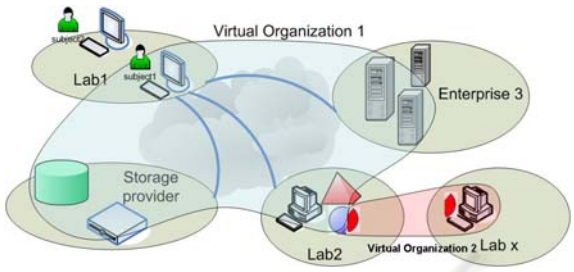

Figure 1: Multiple partners participate in different VOs

The grid is founded on the idea of dispersed resources as well as applications that necessitate acquiring these resources dynamically on demand. This on-demand approach provides tremendous values toward scalability, in addition to aspects of enhanced reusability. The resources are heterogeneous (devices, HD, CPU, clusters…) and attributed by multiple organizations. Take for the example in Figure 1, a physicist at “Lab1” uses a simulation application at a laboratory “Lab2” and then analyzes the data using certain application using a cluster in “Enterprise3” and finally stores the whole results at a storage provider.

The different organizations in a grid environment, federating into an alliance for a certain goal, put in place a Virtual organization. The Virtual Organization (VO) constitutes the common shared space between the different partners. An organization may participate in several Virtual Organizations for different cooperation purposes (ex. Lab2).

The Virtual organization creation embraces topics as management of shared resources and specification of sharing relationships for precise levels of control over how shared resources are put in place and used.

Creating a VO under a single administration, that is one partner controlling the shared space, is not always acceptable. A partner in a VO may be a rival in another, so delegating him the control and management of local resources is inadmissible.

Each domain has to manage its resources locally taking into consideration the cooperation services’ availability. For example, a storage provider may need to move the data on other storage devices locally for maintenance reasons without referring back to the client.

Excluding a central authority, the VO requires scalable multi-administration by the involved partners.

By multi-administration we don't mean a domain with multiple administrators having the same rights, but rather an environment with different administrators having each partial administration rights within the whole structure (Wedde,2003). Though the grid considers inter domains aspects in virtual organization, it doesn't specify how to put in place such a space. The boundaries of a VO should be defined as well as the responsibilities of each organization within it. An access control model is needed to avoid informal methods' errors and security breaches.

An access control model has to specify the relations between the entities within the VO. Since we are dealing with large scale structure and dynamic resources allocation within multi-administered environment, the access control model should be compliant with that. An access control management model should consider the multi-administration aspect within the virtual organization.

On the other hand, to scale well, subjects and objects within VO should be abstracted. For example, dealing with the role "engineer" avoids addressing individually the engineering department users. The same is needed for the resources because of their large number as well as heterogeneity.

Moreover, such abstraction separates the policy from the physical dynamic infrastructure. Policy is then expressed with abstract entities at the VO level, and enforced locally by mapping it to the available demanded resources.

3 RBAC

From the existing access control models, RBAC (Role Based Access Control) (Sandhu, 1997)(Sandhu,1999) has emerged as an access control model more flexible and easier to manage than the traditional MAC, DAC models (Samarati). RBAC aims to facilitate the security management by introducing the "role" notion. A role represents a function in a system (engineer, administrator...). Each role is associated with a set of permissions (or privileges) which is a set of rights corresponding to the tasks which could be realized by this role. Contrary to traditional models, RBAC doesn't associate privileges directly to subjects but it passes via roles. A subject may have several roles, and roles may be attributed to different subjects. Moreover, a role may have multiple permissions and permission may be associated to different roles.

The role definition reflects the organization structure where the role is defined. Roles can be structured hierarchically to facilitate attributing permissions. However RBAC is unable to tackle the complexity of distributed and decentralized organizations. RBAC doesn't express contextual permissions or interdictions which are important in the grid environment example: *role "engineer" has the right to use application "appl. 1" for 3 hours.*

On the other hand, RBAC doesn't reflect the multi-organizational aspect where each organization controls solely certain parts of the system. In such environment, different administrators get into the scene each having his assets that he only controls. We find RBAC limited for modelling the grid virtual organization.

4 ORBAC (ORGANIZATION BASED ACCESS CONTROL)

In the grid environment each partner has resources that are put in common to be shared by the community. However each partner needs to keep the management of his resources local maintaining the internal structures as confidential as possible and revealing only needed information for the well functioning of the system.

What we need is an access control model that indicates: who can do what in which context. OrBAC (Abou El Kalam,2003) (Cuppens,2003) access control model is proposed for modelling a security policy that is not restricted to static permissions and include contextual rules related to permissions prohibitions and obligations. It aims at introducing an abstraction level that separates access control policy from its implementation.

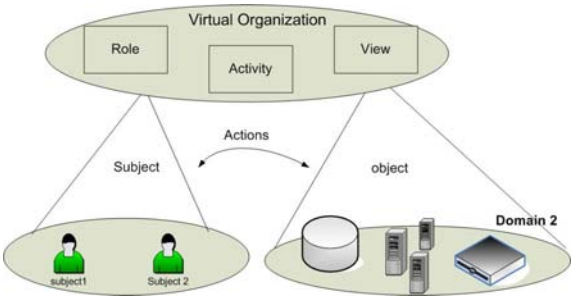

Figure 2: ORBAC access control model

In the grid environment, or the multi-domain environment in general, there are several access stakeholders. The access control model should take into consideration the nature of this multi administered environment.

OrBAC model offers a flexible mean to model such environment employing abstract notions as “role, activities and views” to abstract the traditional “subject action object”.

We recall here the principal notions in this model:

- Organization: it is an organized group of active entities, i.e. subjects, playing some role or other. This notion can be mapped to the Virtual Organization which is the scene of the different activities.

- Role: It corresponds to certain privileges in an organization. The entity Role is attributed to a subject within an organization. Dealing with Roles facilitates users’ management in such a large scale environment. On the other hand, roles are well adapted to our environment where users are dynamic. Policy expressed in terms of roles stay valid even if users change. It is sufficient to assign/revoke a role to a new/departing user.

- View: a view corresponds to a set of properties that can be supplied by a set of objects (storage device, databases, files etc). This notion corresponds well to referencing resources by properties as explained above in the “resources” and “reutilization” issues.

- Activity: It is used to abstract actions as “read", “write". An activity may correspond to a combination of actions.

Or-BAC with these notions responds to the grid requirements. We may consider a Virtual Organization to be an organization in Or-BAC context. Entities in the virtual organization would be roles instead of users, and views instead of objects. The large scale problem of the grid (users, resources) is overcome by the offered abstraction. On the other hand, the multi-administration is also taken into consideration as we will see in the following sections.

5 BASIC COMPONENTS OF ORBAC

There are eight basic sets of entities: Org (organization: an organized group of subjects, playing some role within the group), S (a set of subjects), , A (a set of actions), O (a set of objects), R (a set of roles), $\mathcal{A}$ (a set of activities), $\mathcal{V}$ (a set of views), C (a set of contexts).

OrBAC considers that org $\subseteq$ S, S $\subseteq$ O. Any entity may have attributes, for instance if S is a subject, then name(S), address(S) represents the name and the address of subject S.

Reviewing the relations in OrBAC:

- Empower is a relation over domains Org x S x R. Empower (org, s, r) means that org empowers subject s in role r.

- Use is a relation over domains Org x O x V. Use (org, o, v) means that org uses object o in view v. This helps to give different definitions for the same view.

- Consider is a relation over domains Org x A x A., Consider (org, α , a) means that org considers that actions α falls within the activity a.

- Define is a relation over domains Org x S x A x O x C. Define (org, s, α, o, c) means that within organization org context c holds between subject s, action α and object o.

- Policy definition: access control policy is defined by the relation permission. Permission is a relation over domains Org x R x A. x V x C, Permission (org, r, a, v, c) means that organization org grants role r permission to perform activity a on view v within context c.

We consider that VO is the organization Org in the OrBAC context. To construct this space, add users, attribute users to roles and resources to views and specify access permissions, a management model [Munawer,Cuppens] is needed. It should control the activities: management of organizations, management of roles, activities, views and contexts, assignment of users to roles, assignment of permissions to roles, assignment of users to permissions.

For this mission AdOrBAC (Administration model for OrBAC) (Cuppens,2003) defines views as URA (user/role assignment), PRA (permission/role assignment)... Objects belonging to these views have special semantics, namely they will be respectively interpreted as an assignment of user to role, permission to role. Intuitively inserting an object in these views enables an authorized user to respectively assign a user to a role and a permission to a role.

Conversely, deleting an object from these views will enable a user to perform a revocation.

Distributing the administration functions consists of defining which roles are permitted to access the views URA, PRA … or to more specific views when

the role has not complete access to one of these views.

5.1 URA in AdOr-BAC

This view is used to determine who is allowed to assign a user to a role and on which conditions. Assigning a user to a role equals adding a new object in a given view called URA. Three attributes are associated with this view: subject to designate the subject which is related to the assignment, role that corresponds to the role to which the subject will be assigned and org to represent the organization to which the subject is assigned.

So to add a user to a special role (engineer) in the engineering department eng-dept at organization H, we have to create a view URA-engineer-eng-dept defined as follows:

```
∀ ura, Use (H, ura, URA-engineer-
eng-dept) -> Use (H, ura, URA) ^ role
(ura) = engineer ^ org (ura) = eng-dept
```

This way we defined a view for the engineers in the eng-dept. there is a link between adding an object to this view and the relation empower:

```
∀ Org, ∀ ura, Use (org, ura, URA)
->Empower (org (ura), subject (ura),
role (ura))
```

"Assign" is the acitivity to be given to the administrator to do this assignment job.

Permission (H, administrator, assign, URA-engineer-eng-dept, default)

We may consider the activity "manage" as a combination of two sub-activities (assign , revoke):

```
Permission (H, administrator,
manage, URA-engineer-eng-dept, default)
->
Permission (H, administrator, assign,
URA-engineer-eng-dept, default)^
Permission (H, administrator, revoke,
URA-engineer-eng-dept, default)
```

5.2 PRA in AdOr-BAC

Permission role assignment follows the same logic as URA concerning adding an object to a view; however the object has 5 attributes:

Issuer= issuing organization

Grantee, privilege, target = role, activity and view concerned by the permission

Context = designate the context in which the rule can be applied.

There is a link between adding an object to this view and the relation permission:

```
∀ org, ∀ context, Use (org, pra, PRA)
-> Permission (issuer(pra),
grantee(pra), privilege(pra),
target(pra), context(pra))
```

5.3 VOA in AdOrbac

As in URA (user role assignment), the administrator (ex. Role administrator) adds ura object to the defined view URA. In analogy [9], we would like to define a view including certain collection of objects and attribute the "manage" activity to an administrator (ex. "View-admin")

Associating a user with the role "View-admin"

```
Empower (org, user, View-admin)
```

Attributing permission "manage" to the role "View-admin":

```
Permission (org, View-admin, manage,
VOA-org2, default)
```

Where VOA-org2 is the view of a group of resources having an attribute "org" that is equal to org2, it can be defined as follows:

```
∀ voa, Use(org, voa, VOA-org2) <->
Use(org, voa, VOA)^org(voa)=org2
```

So finally attributing a resource or object to a view in the organization is equivalent to adding a voa with attributes (object to be added, view to which it should be added, organization) to the VOA-org2 view. This can be done by the view-admin.

6 ORBAC MAPPING TO THE GRID

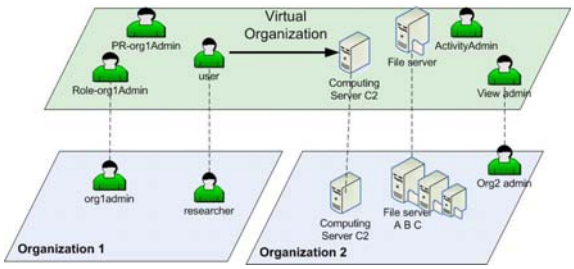

Figure 3: ORBAC model in Grid environment

Consider the scenario in figure 3 where two organizations need to cooperate forming a Virtual Organization. Consider that organization org2 supplies the resources where org1 contains the users.

Putting in place the VO needs to define the participating entities from both sides this includes roles, views, activities and the administration authorities.

Modelling this environment using OrBAC, we start by the virtual organization management itself. Considering that VO is the organization of OrBAC. Then such an entity may have attributes. As discriminating attributes there are the involved participants, a name discriminating a certain VO within the different VOs that can be set up with the same partners, and may be a certain expiry date after which the cooperation is terminated:

```
Name (VO) =cooperation 1
Partners (VO) = org1, org2
Time= deadline
```

For the administration plan, we should define the relevant roles in the Virtual Organization as:

```
Relevant-role (VO, Role-org1Admin);
Relevant-role (VO, PR-org1Admin);
Relevant-role (VO, ViewAdmin);
Relevant-role (VO, ActivityAdmin).
```

Also the Relevant activities and the Relevant views names.

To attribute to a certain "org1admin" the role "*Role-org1Admin*" that has the responsibility to assign/revoke roles to the users of organization org1:

Empower (VO, org1admin, Role-org1Admin)

Permission (VO, roleadmin, manage, URA-org1,default)

Where URA-org1 is defined as:

```
∀ Ura, Use (VO, ura, URA-org1) <=>
Use (VO, ura, URA) ^ org (ura) = org1
```

To attribute to a certain "org1admin" the role "PR-org1Admin" that has the responsibility to assign/revoke permissions to roles of organization org1:

```
Empower (VO, org1admin, PR-org1Admin)
Permission (VO, PR-org1Admin, manage,
PRA-org1, default)
```

Where PRA-org1 is defined as:

```
∀ pra, Use (VO, pra, PRA-org1) <=>
Use (VO, pra, PRA) ^grantee(pra)∈ {r/ ∃
ura∈ URA-org1, role(ura)=role}^
privilege (pra) ∈ activities(VO)
```

To attribute to a certain "org2admin" the role "ViewAdmin" that has the responsibility to assign/revoke views to objects of organization org2:

```
Empower (VO, org2admin, View-org2Admin)
Permission (VO, View-org2Admin, manage,
VOA-org2, default)
```

Where *VOA-org2* is defined as:

```
∀ voa, Use (VO, voa, VOA-org2) <=>
Use (VO, voa, VOA) ^ org (voa) = org2 ^
view(voa) ∈ views(VO)
```

To attribute to a certain "org2admin" the role "ActivityAdmin" that has the responsibility to assign/revoke activities to actions comprehensible by organization org2 and its different resources:

```
Empower (VO, org2admin, ActivityAdmin)
Permission (VO, ActivityAdmin, manage,
AaA-org2, default)
```

Where *AaA-org2* is defined as:

```
∀ aaa, Use (VO, aaa, AaA-org2) <=>
Use (VO, aaa, AaA) ^ org (aaa) = org2
```

Assigning concrete level to the abstract level entities (of course done by authorized roles/users as indicated in the administration plan above):

Role-org1Admin:

```
Empower(VO, Rlocal1, Rvo1)
Empower(VO, Rlocal2, Rvo2)
```

ViewAdmin:

```
Use(VO,storagedevice,Objlocal1&Objlo
cal2)
Use(VO,applicationserver, Objlocal1)
```

ActivityAdmin:

```
Consider (VO, action1, Activity1)
Consider (VO, action2, Activity2)
Consider (VO, execute, Execution)
Consider (VO, write, Update)
Consider (VO, read&write, Modify)
```

PR-org1Admin:

```
Permission(VO,Rvo1,Update&activity1,
storage device, workTime)
Permission(VO,Rvo2,Execution&Modify,
application server, day&night)
```

7 CONCLUSION AND FUTURE WORKS

The grid has a goal enabling coordinated resource sharing and problem solving in dynamic multi-domain virtual organization. For this reason it sets up a virtual organization which constitutes the

shared common space between the different partners.

On the way to dynamically create virtual organizations we needed an access control model that takes into consideration the grid environment constraints as inter-domain aspect, large scale, dynamic resource allocation…. We presented OrBAC as the candidate having too many features appropriate for this mission. OrBAC models a multi-administered environment using the "role view activity" abstraction which abides to the Grid constraints (large scale and dynamic resources).

However, since the dynamic resources allocation can be propagated on multiple sites on behalf of the user, the model should realize delegation aspects (Welch, 2004). Delegation will be discussed in details in another paper. After completing the model an analysis is needed to study the appropriate way for implementing OrBAC within the different Grid layers (Foster, 2001).

REFERENCES

Alfieri R., Cecchini R., Ciaschini V., dell'Agnello L., Frohner A., Gianoli A., L˝orentey K., and Spataro F. (2003). VOMS, an authorizatuin system for virtual organizations. Presented at *1st European Across Grids Conference, Santiago de Compostela, February 13-14, 2003.* http://grid-auth.infn.it/docs/VOMS-Santiago.pdf

Abou El Kalam A., El Baida R., Balbiani P., Benferhat S., Cuppens F., Deswartes Y., Miege A., Saurel C., Trouessin G. (2003). "Organization Based Access Control". In Proceedings *of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks (Policy'03), p.120-131, June 4-6, 2003, Lake Como, Italy.*

Cannon S., Chan S., Olson D., Tull C., Welch V., Pearlman L. (2003). Using CAS to manage Role based VO sub-groups. In *CHEP 2003. La Jolla, California.*

Baru C., Moore R., Rajasekar A., Wan M. (1998). The SDSC Storage Resource Broker. In *Proc. CASCON'98 Conference, Nov.30-Dec.3, 1998, Toronto, Canada.* http://www.npaci.edu/DICE/Pubs/CSI-paper-sent.doc

Djordjevic I., Dimitrakos T., Phillips C. (2004). An Architecture for Dynamic Security Perimeters of Virtual Collaborative Networks. In *Proceeding 9th IEEE/IFIP Network Operations and Management Symposium, (NOMS 2004), April 2004. IEEE-CS.*

Cuppens F., Miege A. (2003). Ad-OrBAC: An Administration Model for Or-BAC. *Workshop on Metadata for Security, International Federated Conferences (OTM'03), Catania, Sicily, Italy, November 3-7, 2003.*

Foster I., Kesselman C., Tuecke S. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In, *International J. Supercomputer Applications,15(3),2001.*

Foster I., Kesselman C., (1997). A Metacomputing Infrastructure Toolkit. *Intl J. Supercomputer Applications, 11(2):115-128.*

Fedak G., Germain C., Neri V., and Cappello F. (2001). XtremWeb: A Generic Global Computing System. *CCGRID2001, workshop on Global Computing on Personal Devices, May 2001, IEEE Press.*

Karl Czajkowski, Ian Foster, Carl Kesselman, Volker Sander, Steven Tuecke, (2002). SNAP: A Protocol for Negotiation of Service Level Agreements and Coordinated Resource Management in Distributed Systems. *Draft submission to JSSPP'02 April 30, 2002.* Retrieved January 26, 2005 from:
http://www-unix.mcs.anl.gov/~schopf/ggf-sched/GGF5/sched-GRAAP.3.pdf

Nitin Nayak, Tian Chao, Jenny Li, Joris Mihaeli, Raja Das, Annap Derebail, Jeff Soo Hoo, (2001). Role of Technology in Enabling Dynamic Virtual Enterprises. Retrieved January 26, 2005 from:
http://cersi.luiss.it/oesseo2001/papers/13.pdf

Samarati P., De Capitani di Vimercati S.. Access Control: Policies, Models, and Mechanisms. Retrieved January 26, 2005 from: http://www.ic.unicamp.br/~rdahab/cursos/inf712/material_didatico/docs/LNCS2171_Cap3.pdf

Russell D., Dew P., Djemame K (2004). Access control for dynamic virtual organizations. In *Proceedings of the UK e-Science All Hands Meeting 2004, © EPSRC Sept 2004*. Retrieved January 26, 2005 from: http://www.allhands.org.uk/2004/proceedings/proceedings/proceedings.pdf

Sandhu R., Coyne E., Feinstein H., Youman C. (1996). Role-Based Access Control Models. *IEEE Computer, vol. 29, n° 2, pp.38-47, février, 1996.*

Sandhu R., Munawer Q. (1999). The ARBAC99 Model for Administration of Roles. In *Proceeding of the 15th Annual Computer Security Applications Conference (ACSAC'99), Phoenix, Arizona, 6-10 December 1999, IEEE Computer Society, pp. 229-241.*

Welch V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S. and Siebenlist, F. (2004). X.509 proxy certificate for dynamic delegation. *Proceedings of the 3rd Annual PKI R&D Workshop.*

Wedde H.F., Lischka M., (2003).Cooperative Role-Based Administration. *Proceedings of the eighth ACM symposium on Access control models and technologies Como, Italy 2003*