УДК 681.3.07

K. Matusiewicz, magister, Catholic University of Lublin, Poland;
P.P. Urbanowich, professor

# MANAGEMENT OF DATA OBJECTS IN ACCESS CONTROL MODELS BASED ON ROLES

**Abstract:** Management of access rights to objects is one of the fundamental duties of access control mechanisms. The concept of object grouping for access control management and formally describe new access control model utilizing that idea are proposed.

## 1. Introduction

Present computer systems have to deal with great amount of data objects shared between many users under diversity of security policies. Managing access rights in such environment is either too loose, and therefore potentially risky, like in the case of DAC [8,9], or too limited for most of cases, like in mandatory access control [1, 3, 4].

That is why hierarchical models of access control, like RBAC [12], were developed. In that approach, flat management of single access rights was replaced by management of collections of rights, called roles. Each role represents a set of rights necessary for performing certain tasks. Users designated to perform a task are assigned appropriate set of roles and that way they acquire necessary access rights which are included in roles they possess. This situation is depicted in Fig. 1.

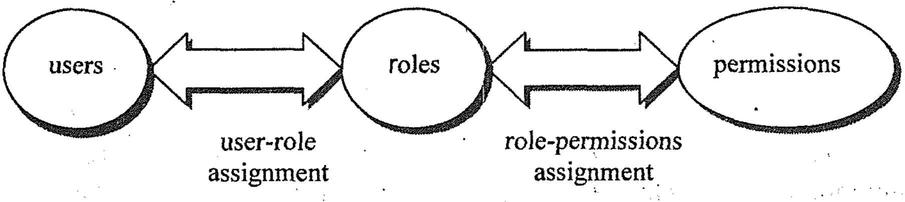

Fig. 1. Relationship between users, roles and permissions in RBAC.

So far, management of roles attracted considerable attention and much has been done in that direction. Role hierarchies, administrative roles [11], constraints like static and dynamic separation of duties [5,6] are just a few examples of important improvements in role management.

Comparing to that, there is a lack of mechanisms simplifying management of data objects. In this paper we will propose one solution to that problem by means of object grouping.

## 2. Object grouping in access control mechanism

There are two characteristic features of data objects in modern information systems. The first is the number of them, which is counted in thousands. The second is that many objects share the same origin or purpose, so, from the point of view of access control mechanism, users have the same access rights to the whole group of objects. Good examples are files in user's home directory or files of WWW service.

These observations lead to the concept of domains. A domain is a set of objects in the system. There are no restrictions on kind of objects which are members of a domain, but natural approach is to group together objects which are connected by the usage or purpose. The concept of domains grouping objects is shown in Fig. 2.

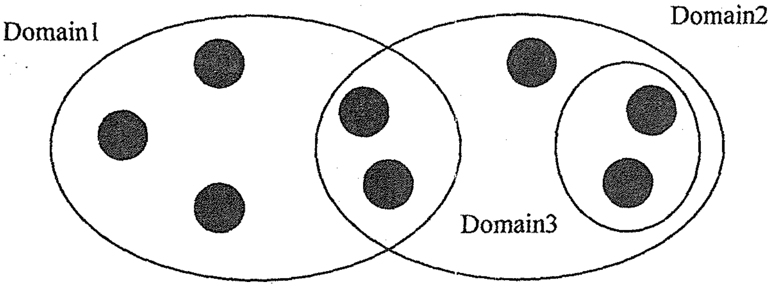

Fig. 2. Domains grouping data objects

It is worth to note, that well-known mechanism of files inheriting access rights from parent directory is a special case of domain approach. Files are data objects and directories play role of domains gathering files together and providing way of managing access rights for the whole set of them. However, this approach has limitations which are not present in general domain–based model. Directories form a tree–like structure and therefore a single file is contained in precisely one of them. In case of domains, one data object can be a member of many domains, so access rights can be multiple inherited from each domain. Besides of that, primary purpose of directories is organizing storage of files rather than managing security. Separation of file grouping from directory structure allows for manipulation in the same time of rights of files which have to be in different locations.

When domains are defined in system, access rights are not connected with single objects anymore but rather with domains, as shown in Fig. 3.

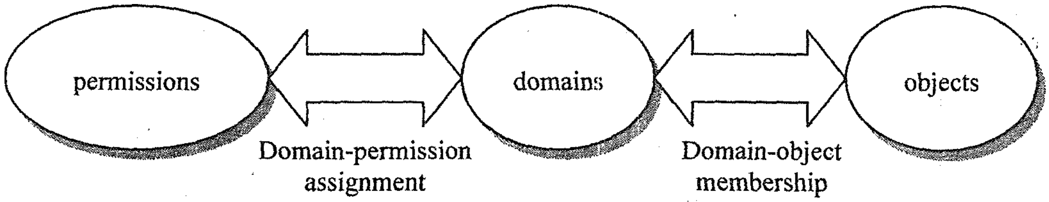

Fig. 3. Relationship between permissions, objects and domains

Now managing of access rights to data objects consist of following activities, which can be done separately:

1) adding objects to domain,
2) removing objects from domain,
3) adding access rights to domain,
4) removing access rights from domain.

That simplifies the task of object management and also allows for more flexible approach, because above activities can be performed separately, perhaps by different individuals or mechanisms, so more sophisticated access control policies can be expressed using the concept of domains. The idea of object grouping is very general and can be a base for many specific solutions according to particular needs. One of them will be proposed below as an example.

## 3. Access control model with domains

In this section we define in formal way access control model which connects two mechanisms important for simplifying work with access control, roles and domains. Roles are used for permission grouping and domains for object gathering.

### 3.1 Access control components

The basic elements of proposed access control model are *users*, *subjects*, *objects* which are used for representing external entities, and *rights*, *roles* and *domains* used for expressing security policy.

**Users** represent people or autonomous mechanism which use the system. The set of users will be denoted by $\boldsymbol{U}$.

**Objects** are passive elements of system, which store the information. The set of objects will be denoted as $\boldsymbol{O}$.

**Subjects** are active elements of system. They act on behalf of users, process information taken from data objects and store the results in other data objects. Subjects can be identified with running processes. The set of subject in the system will be denoted as $\boldsymbol{S}$.

Possible operations which can be performed on data object by subject are described by **access rights**. Usually there is reading, writing, creating and deleting among them, but the set of rights depends on the characteristics and purpose of the system. In our case the set of rights is divided into two disjoint subsets, the set of *file rights* and the set of *administrative rights*. The set of file rights describes possible file operations and is denoted as $\boldsymbol{P_F}$. Set of administrative rights, denoted by $\boldsymbol{P_A}$, includes rights provided for modification of access control mechanism structures, like creating domain, creating user or assigning permissions to role.

**Roles** are used for grouping permissions necessary for performing specified tasks. The set of roles in the system will be denoted by $\boldsymbol{R}$.

**Domains** are used for grouping of objects for common access rights management. A domain is simply a distinguished subset of data objects.

### 3.2 Relations between components

The core of the model is the set of relations between elements constituting the model. They are defined by the following functions.

Each user can have a subset of roles assigned. It is described by the function

$$\rho : U \to 2^R,$$

which assigns to the user *u* the subset of roles for that user.

Symmetrically, each object is a member of a certain number of domains. It is described by the function

$$\delta : O \to 2^D,$$

which for each object *o* returns the subset of domains.

Existence of each subject is initiated by certain user, so for each subject we can find the user who is "responsible" for that subject. It is done by the means of the function

$$c:S\to U,$$

which returns the user who owns given subject.

The most crucial role for definition of access control model plays the function which for given pair *(role, domain)* returns the subset of file rights. It is function

$$h_F:R\times D\to 2^{P_F},$$

which for given role, describes the set of rights for specified domain present in that role. In the similar way we define function which for each role defines the subset of administrative rights. Administrative rights are not connected with specified domain, but are universal, so the function returning subset of administrative rights for given role is the function

$$h_A:R\to 2^{P_A}.$$

User can have many roles assigned and one object can be in many domains, so it is necessary to define function operating on subsets of roles and domain and returning access rights. The most natural approach is to sum rights over or roles and domains, what can be described by the following function

$$H_F:2^R\times 2^D\to 2^{P_F},$$

defined for $U\subseteq R$ and $B\subseteq D$ as

$$H_F(U,B)=\bigcup_{r\in U}\bigcup_{d\in B}h_F(r,d).$$

With the help of described above formulae, we can finally express access control function

$$A:S\times O\to 2^P,$$

which for given subject $s$ and object $o$ returns subset of access rights. It is defined as

$$A(s,o)=H_F(\rho(c(s)),\delta(o)).$$

Subject $s$ has file rights for the object $o$ which are the sum over all roles assigned to subject's owner to domains containing the object.

Additionally, each subject can have certain number of administrative rights depending on the set of roles assigned to user on behalf of which subject works. That situation is defined by the function

$$Ar(s)=\bigcup_{r\in\rho(c(s))}h_A(r).$$

State of access control mechanism is therefore defined by the tuple $(U,S,O,R,D,A,Ar)$ together with functions $\rho,\delta,c$. Schematic illustration of model's elements and relationships between them is shown in Fig. 4.

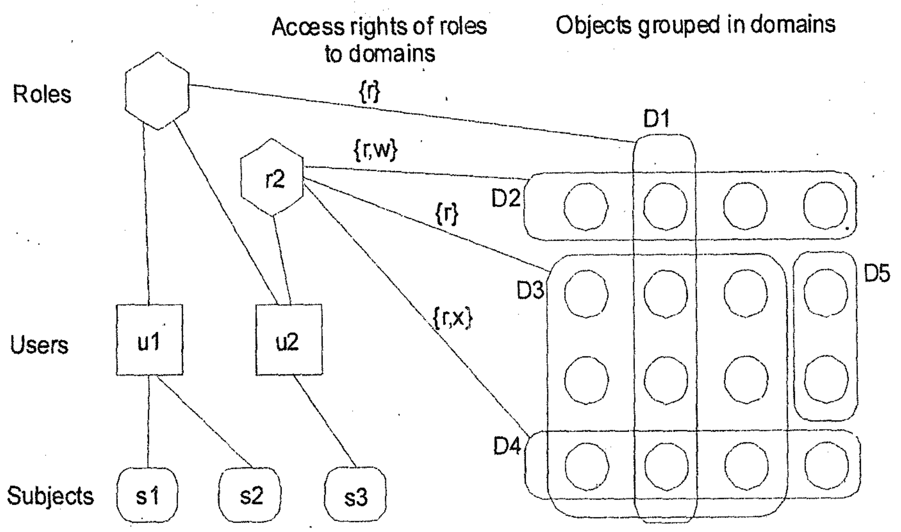

Fig. 4. Illustration of proposed access control model

## 4. Model analysis and implementation considerations

Described model is an extension of features of well-known RBAC model. When for each object there is defined single domain containing that object, the model reduces to classical RBAC system. In our model, there is no difference between simple and administrative roles as described for example in [9], because each role can have administrative rights. In the sense of expressive power, these approaches are equivalent, because each role in proposed model can be simulated by two RBAC roles: simple and administrative, and each RBAC role, either simple or administrative, can be expressed as single role in our model.

This model was implemented as a part of work [10] in the networked file sharing system and now is being tested. Object-oriented implementation in Java [7] was chosen, where elements of model were mapped directly to classes. There were no difficulties in performing that operation, so it seems that defined formally model is suitable for practical applications. It also appeared that described above state of access control mechanism have elegant and compact representation in XML [2].

## 5. Conclusions and future work

In this paper we presented the idea of object grouping for simplifying security management and proposed improved model of access control based on roles using notion of object domains. Thanks to that, managing of rights to data objects becomes simpler and more intuitive task.

Proposed access control model does not have any sophisticated solutions in order to be as general as possible and sets up a framework rather than a particular solution. Thanks to that it can be easily modified and upgraded with new features like negative (inhibiting) rights, role inheritance and perhaps constraints, depending on the needs of security policy.

Future research will be concentrated on adopting that model for distributed access control, where there is no single access control authority but a group of cooperating centers.

# BIBLIOGRAPHY

1. D.E. Bell, L.J. LaPadula. *Secure Computer Systems: Mathematical Foundations* MTR-2547 (Vol. I, II), MITRE Corp., Bedford, MA, 1973.

2. T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler (eds.) *Extensible Markup Language (XML) 1.0 (Second Edition)* W3C Recommendation, October, 2000.

3. D.E. Denning. *A lattice model of secure information flow* Communications of the ACM, vol. 19 no. 5. May, 1976.

4. D.E. Denning, T.F. Lunt, R.R. Schell, M. Heckman and W. Shockley. *The SeaView Formal Security Policy Model* SRI Interim Report A003, SRI International, 1987.

5. S.I. Gavrila, J.F. Barkley. *Formal Specification for Role Based Access Control Uer/Role and Role/Role Relationship Management,* Proc. of 3rd ACM Workshop on Role–Based Access Control, Fairfax, Virginia, October, 1998.

6. L. Giuri, P. Iglio. *A formal model for role-based access control with constraints,* Proc. of 9th IEEE Computer Security Foundations Workshop, Kenmare, Ireland, June 1996, pp. 136—145.

7. J. Gosling, B. Joy, G. Steele. *The Java Language Specification,* Addison Wesley Developers Press, Sunsoft Java Series, 1996.

8. G.S. Graham, P.J. Denning. *Protection – principles and practice,* Proc. Spring Jt. Computer Conf., vol.40, AFIPS Press, Montvale, N.J. 1972.

9. M.A. Harrison, W.L. Ruzzo, J.D. Ullman. *Protection in Operating Systems*, Communications of the ACM, vol. 19 no.8, 1976.

10. K. Matusiewicz. Modele kontroli dostępu do zasobów i ich wpływ na bezpieczeństwo systemów komputerowych, M. Sc. thesis, Catholic University of Lublin, Poland, 2002

11. Q. Munawer *Administrative Models for Role–Based Access Control*, PhD dissertation, George Mason University, 2000.

12. R. Sandhu, E.J. Coyne, H.L. Feinstein, Ch.E. Youman. *Role–Based Access Control Models*, IEEE Computer, Vol. 29, No. 2, February, 1996.