Role engineering of information system using extended RBAC model

Aneta Poniszewska-Maranda
Institute of Computer Science,
Technical University of Lodz, Poland
anetap@ics.p.lodz.pl

Abstract

The role-based access control (RBAC) model is one of the policies used to access control in information systems for enterprises. The RBAC model is a powerful technology for managing and enforcing security in large-scale, enterprise-wide systems. Many implementations of this model, including the RBAC96 model, have been already proposed.

This paper presents an extension of the standard RBAC model together with its implementation using the Unified Modeling Language (UML). The presented model is developed for the role engineering in the security of information system.

In the paper the union of the RBAC model, which controls access in the information system, and the UML language, i.e. a unified method of object analysis and design, is proposed. The presented approach of the RBAC model consists in role creation via defining appropriate permissions. The entire procedure is performed in two stages; first permissions assigned to a function are defined, and then definitions of functions assigned to a particular role are provided.

1. Introduction

In information systems the data protection against improper disclosure or modification is an important requirement. The access control regulates what a user can do directly and what the programs executed on behalf of the user are allowed to do. The system administrators have to implement access control mechanisms to protect the confidentiality and integrity of applications and its data. In the past, the user access was granted by adding necessary permissions to each individual application, which made administration involving many users and several different applications difficult and ineffective.

The alternative is not to assign users directly to permissions for each application but to assign users to roles and the permissions to roles for each application. Any change in the user's position or their needs can be simply solved by assigning the user to another role. The role-based access control (RBAC) model was chosen to present and realize the access control in the security processes of an information system. The RBAC model allows the administrator to assign users to roles rather then users directly to permissions. The most important concept of the RBAC model is that of a role. This concept facilitates the access control management performed by the security administrator since users and permissions can be assigned to role. Each user has a utilization profile that defines his roles in the enterprise. The role is the set of permissions that allow execution of the system functions and the access to the data used by these functions. The role is properly viewed as a semantic construct around which the access control policy is formulated. Recent research efforts focus on different aspects of role engineering using RBAC models [4, 5]

The methods of the object-oriented analysis and conception appeared in the 90-ties as an evolution towards greater coherence among the information system objects and their dynamics. The unified method - Unified Modeling Language (UML), which can be mentioned in this category, was created by J. Rumbaugh, G. Booch and I. Jacobson, and is the fusion of the three methods presented by these authors [2, 6]. The UML has been chosen for the representation and implementation of extended RBAC model because nowadays it is a standard tool, properly reflecting the description of the information system and its needs. UML has become a standard also for the object-oriented modeling in the field of the software engineering.

This paper presents the process of role creation in the security approach of an information system. It shows a possibility of implementing the extended RBAC model by means of the UML language. The first part of the paper describes our extension of the standard RBAC model [8, 10] and the representation of this extension using the UML concepts.

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE

The second part deals with the role engineering in the information system security - the implementation of extended RBAC model.

2. Extended RBAC model

The role-based model regulates the access of users to information basing on activities that the users perform in a system [8, 9, 10]. It requires identification of roles in this system. In the course of study of this issue, the standard model was extended by addition of new elements [3, 7] (Figure 1).

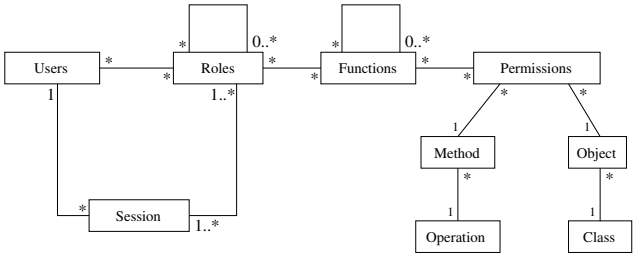

Figure 1. Extension of the RBAC model

Each role defined in the RBAC model realizes a specific task in the enterprise process. A role contains many functions that the user can take. For each role it is possible to choose the system functions which are necessary for it. Thus a role can be presented as a set of functions that this role can take and realize. Each function can have one or more permissions, and consequently a function can be defined as a set or a sequence of permissions. If an access to an object is required, then the necessary permissions can be assigned to the function to complete the desired job. All the tasks and required permissions are identified and they can give the possibility to perform all the responsibilities of the role.

Decomposition of roles can be exemplified by an application in a university information system. The user *professor* can have two roles: *Teacher* and *Researcher* to perform the teaching and researching. These roles contain some functions which are mapped to the sets of permissions that grant the access to perform the jobs required by each function (Figure 2).

Specific access rights are necessary to realize a role or a particular function of this role. These rights determine the possibility to execute an application or to access necessary data, and moreover they correspond to these functions. Thus specific permissions have to be defined for each function. In an object oriented approach the permission determines the right to execute a particular method on a particular object [3]. It shows that in order to access data stored in an object a message has to be sent to this object. This

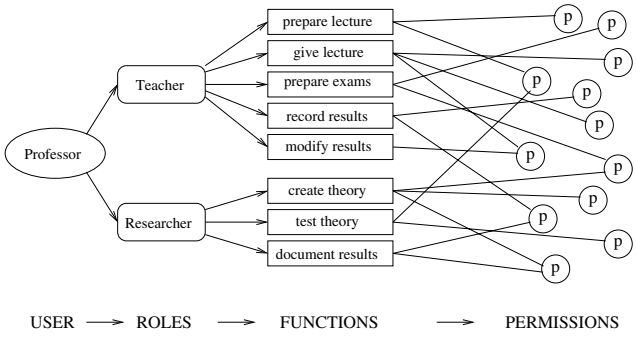

Figure 2. Example of the role-function-permission mapping

message causes execution of a particular method on this object. The same permission is defined for all instances of the object except the contrary specification. A constraint determines that some permission is valid only for part of the instances of the object. Consequently, *a permission is defined by a function* ***p(m, o)****, where* ***m*** *is a method that can be executed on a set of object instances* ***o*** *and the set* ***o*** *can be determined by a set of constraints concerned by this permission.*

The permission in this model represents a positive authorization - if a permission exists for an object it means that some method can be executed on this object.

2.1. Representation of the extended RBAC model using the UML concepts

The Unified Modeling Language (UML) is a high-profile approach to modeling system [2, 6]. Being a standard modeling language in the area of the software engineering, the UML has also become a support for object-oriented approaches. It contains a suite of diagrams for requirements, analysis design and implementation phases of the development of systems.

Two types of UML diagrams have been chosen to implement the extended RBAC model: use case diagram and sequence diagram. The use case diagram represents the functions of the system from the users' point of view [2]. It determines the behavior of the system without defining the internal structure of the functions. According to the definition of the UML meta-model, each use case diagram contains some use cases and each use case represents a collaboration, i.e. a scenario. To determine each collaboration an interaction diagram can be created.

The interaction diagram describes the behavior of one use case [2]. It represents the objects and messages exchanged in the use case. There are two types of interaction diagrams: sequence diagram and collaboration di-

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE

agram. Both of them are defined to capture how objects collaborate with each other to realize the use cases.

The UML language gives the possibility to present the system using different models. The purpose of the presented study is to implement and realize the extended RBAC model with the use of UML. To aim this, the conceptions of the UML language and extended RBAC model should be first joined (Figure 3). The presentation of the connections between the UML and RBAC concepts is given in [7]:

- RBAC role is joined with UML actor,
- RBAC function joined with UML use case,
- RBAC methods and objects with UML methods and objects,
- RBAC permissions can be found in the interaction diagrams,
- RBAC constraints are joined with the constraint concept existing in the UML,
- relations of different types that occur between the elements of the RBAC model can be found in the use case diagrams and in the interaction diagrams.

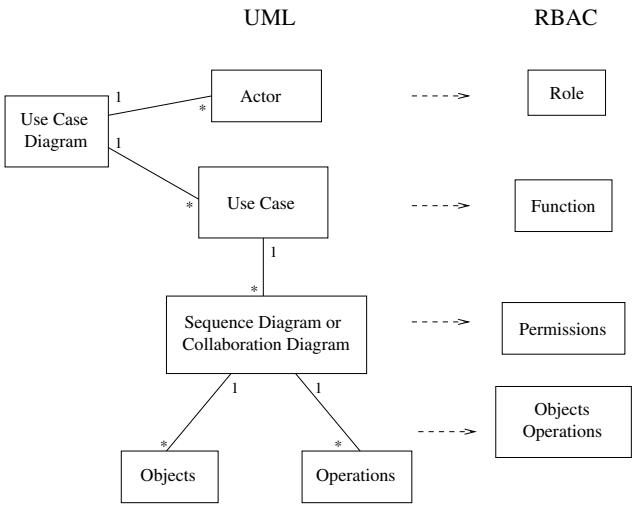

Figure 3. UML concepts and their relationships with the RBAC model

3. Role engineering — implementation of extended RBAC model

System security policies demand that for each user a set of operations that he will be allowed to execute should be clearly defined. Consequently, for each user a set of permissions should be defined. It suffices to specify the permissions for execution of certain methods on each object accessible for that user. This process is realized with the use of sequence diagrams, where permissions are assigned to the rights of execution of the methods realized in each use case.

Basing on the connection between UML and extended RBAC model given above, a definition of a security system access control (of RBAC type) on the developer level is proposed, using the UML language as a tool. The UML metamodel is applied to define the roles of RBAC models, the functions that are used by these roles to co-operate with the information system and the permissions needed to realize these functions. Owing to use case diagrams a list of actors co-operating with the information system is obtained. An analysis of these diagrams allows automatic specification of relations of the following types: R-R (role-role) relation (with the use of generalization relation between the actors), R-F (role-function) relation (with the use of the association relation between the actors and the use cases) and F-F (function-function) relation (generalization relation between the use cases).

The description of a use case using the interaction diagrams (e.g. sequence diagrams) presents activities needed to realize the functions of a system. Each activity is a definition of execution of a method on an object. It is possible to add the permission *execute* to each method attached to message sending in a sequence diagram and next make an union with the set of permissions. Therefore the F-P (function-permission) relations can also be automatically managed.

3.1. Rules of role creation

In our study, the process of access control realization in information system, i.e. the process of role engineering, is divided into two levels:

- level of application developer on witch the roles of an information system are determined,
- level of security administrator who defines the associations between the roles and system users.

An user is assigned to a user profile witch is defined by the set of roles played by him. A user profile is defined by a pair *(u, listRoles)*: *u* is a user, *listRoles* is a set of roles assigned to this user.

It is possibly to give the rules of role creation and assignments of these roles to users:

1. A profile should be defined for each user who interact with the system

$$u_i \vdash userProfile_i$$

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE

2. A profile is defined by a set of roles which a user can take

$$userProfile_i \vdash setRoles_i$$

To receive a significative profile, each user should be assigned at least to one role.

3. A role is defined by a set of functions assigned to this role

$$r_j \vdash setFunctions_{r_j}$$

To receive a significative role, each role should have at least one function assigned.
In UML description, each actor should be assigned at least to one use case in the use case diagram

$$a_j \vdash setUseCases_{a_j}$$

4. A function is defined by a set of permissions necessary to it execute

$$f_k \vdash setPermissions_{f_k}$$

To receive a significative function, each function should have at least one permission assigned.
In UML description, using the interaction diagram (i.e. sequence diagram or collaboration diagram) each use case should be defined by detail description, i.e. represented by a set of methods executed on the objects

$$uc_k \vdash setPrivileges_{uc_k}$$

3.2. Creation of roles

Our definition of a set of roles in an information system with the use of UML diagrams contains two stages:

- assignment of a *set of privileges* (permissions) to an *use cas*e in order to define the function and
- assignment of a set of *use cases* (functions) to an *actor* in order to define the role.

3.2.1 Definition of permissions assigned to a function

A use case, which is a description of activities and needs of an actor, allows specification of the actor's interactions and co-operations. Interactions between the objects are represented by a sequence of actions that allow definition of a set of privileges. Therefore, in order to identify the permissions assigned to the function it is necessary to start from the sequence diagram.

Thus for each use case one should define a set of permissions needed to activate the represented function. The interaction is defined as a set of messages and each message is realized by an action that can change the state of the system. For each message of the sequence diagram, one has to assign a permission that gives the right to its execution.

Each message $msg\,(o_1, o_2, m)$ in the interaction sent by object o_1 to object o_2 to execute method m on object o_2 should be assigned to a suitable permission. This permission corresponds to the possibility of the execution of method m on object o_2. On the addition of constraints, the definition of the message is as follows: $msg\,(o_1, o_2, m, cst)$ where cst represents a set of constraints. Consequently, the set of permissions for interaction i is defined as follows:

$$P\,(i) = \{p \mid \varphi\,(msg\,(o_1, o_2, m, cst)) =$$

$$p\,(m, o_2) \wedge cst\,(p) = true\}$$

where φ is a function that assigns a permission to message msg and o_1 is an actor or class instance that can execute method m on object o_2.

The interaction diagram, in particular the sequence diagram, is defined by the set of interactions. Thus, the set of permissions assigned to diagram d and described by the set of interactions D is:

$$P\,(d) = \bigcup_{i \in D} P\,(i)$$

The use case μ described by set M of interaction diagrams d_i has the following set of permissions assigned to it:

$$P\,(\mu) = \bigcup_{d_i \in M} P\,(d_i)$$

The set $P\,(\mu)$ represents set of permissions assigned to the function specified by use case μ.

3.2.2 Definition of functions assigned to a role

The use case diagram allows visualization of the set of functions of a system by examining the needs of each actor. Thus, the use case diagram representing relations between the actors and the use cases and specifying the set of functions that should be assigned to the role, becomes the starting point.

The use case diagram dcu_i contains the use cases (functions) joined with some actors (roles). The set of functions assigned to role r, described by one use case diagram, is defined by the functions that are in a direct or indirect relation with this role (by the heritage relation between the functions) on this diagram:

$$F\,(r_{dcu_i}) = \{f \mid f = uc\ for\ uc \in dcu_i \wedge (r_{dcu_i}, f)$$

$$\in R - F\} \cup \{f' \mid f' = uc\ for\ uc \in dcu_i \wedge$$

$$((f, f') \in F - F \wedge (r_{dcu_i}, f) \in R - F)\}$$

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE

The set of functions of the role is defined by the union of use cases assigned to this role in all use case diagrams D_{cu_i}:

$$F(r) = \bigcup_{dcu_i \in D_{cu}} F(r_{dcu_i})$$

In order to create a set of roles assigned to a user profile, users should be assigned to roles. The security administrator is responsible for this stage. It should be emphasized that there is an essential difference between the role management performed by [3]: the developer of the information system who determines the roles of an application and the security administrator who assigns users to the roles in the information system.

3.3. Constraints

The application developer has a good knowledge of a given application and can describe security rules for that application. On the other hand, only the global administrator, who knows the security policy of the organization, can describe properly the global security of a system. On each of two levels a set of necessary security constraints should be formulated. The following classification of constraints in security realization of information system is proposed [3]:

- constraints from the developer's point of view - constraints at the application level, specified to create a complex application,
- constraints from the administrator's point of view - constraints at the global security level of an enterprise.

Conception level is realized by the application developer who creates an application using UML. We can distinguish the following types of his constraints:

- constraints on a permission: static - constraint reduces a set of objects accessible through a particular method independently of the user who executes the method, e.g.: the permission of "write" the documents with name *.doc; dynamic - constraint reduces a set of objects accessible in the course of a session (in the course of an execution that needs a particular access mode in a system), e.g.: the permission of "write" the documents for a user who is the owner of these documents,
- prerequisite constraints - based on the prerequisite concept: a permission p can be assigned to a function only if this function has a permission q, e.g.: the permission of "read" some file asks the permission of "read" the directory in which this file is situated,
- cardinality constraints - numerical limitation on application elements, e.g.: the permission of "read" the file with confidential information about the system users can be given only to one specific role.

On the exploitation level the security administrator manages a set of applications and gives the system users different rights to use particular applications. The most important constraints on this level can be classified as follows:

- separation of duty (SOD) constraints - represent the concept of mutually exclusive roles and mutually exclusive permissions [1],
- prerequisite constraints - based on the concept of a prerequisite role; e.g. a user can be assigned to role Teacher only if he has been assigned to role Employee,
- cardinality constraints - numerical limitation on classes in role-based system; e.g. the number of users authorized to be assigned to a role is limited; in application example there is only one person who can be assigned to role Dean.

Other types of constraints concern the role hierarchy or session concept in RBAC model.

The OCL (Object Constraint Language) [11] is proposed as the most suitable language for the formulation of the developer's constraints because it was created to allow constraint definitions in different UML diagrams. On the other hand, the use of RCL 2000 (Role-Based Constraints Language 2000), created to support the constraints of RBAC model, seems to be the best to express the administrator's constraints. OCL is an expression language which gives the possibility to define the constraints in object-oriented models. OCL expression can be used in each UML diagram to specify a condition, a specific object or a set of objects. Invariants and preconditions in class diagrams and sequence diagrams have been chosen as effective tools for presentation of our security constraints. It is possibly to translate the constraints of developer level, expressed in OCL, to the RCL 2000, to have the set of all security system constraints formulated in one language created to support the security constraints. We defined the set of rules for this translation but its presentation does not concern the subject of this paper.

4. Conclusions

The paper proposes the process of role engineering in the information system security using extended RBAC model. The presented process is based on implementation of the extension of classic RBAC model using the UML language. The UML, a standard language of object analysis and design nowadays, has been chosen in view of the fact that it enables complex presentation of the information system and of important aspects of the information system security in particular. On the other hand, the access control based on

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE

roles allows simple management of information system for security administrator.

The paper presents the relationships between the extended RBAC concepts and those of the UML language. The elaborated RBAC extension gives the possibility to manage the system and to make the process of the role creation simpler, particularly in the conception phase. The implementation of extended RBAC model is also based on the UML. The creation of the user profile, performed in two stages: definition of permissions assigned to a function and definition of functions assigned to a role, is realized using the UML concepts connected earlier with the concepts of the extended RBAC model.

Our next research is concentrated on access control constraints of the information system, the constraints of extended RBAC model in particular, as well as on their implementation on UML-Java platform containing XML notation and developed for extended RBAC model [3].

References

[1] G.-J. Ahn and R. S. Sandhu. Role-based authorization constraints specification. *ACM Transactions on Information and Systems Security*, 2000.

[2] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison Wesley, 1998.

[3] G. Goncalves, F. Hemery, and A. Poniszewska. Verification of access control coherence in information system during modifications. *Proc. of the 12th IEEE WETICE Workshops, Austria*, 2003.

[4] Q. He and A. I. Anton. A framework for modeling privacy requirements in role engineering. *Proceedings of the 9th International Workshop REFSQ, Austria*, 2003.

[5] G. Neuman and M. Strembeck. A scenario-driven role engineering process for functional rbac roles. *Proceedings of the 7th ACM Symposium SACMAT, Monterey*, 2002.

[6] Object Management Group. *Unified Modeling Language Specification, Reference Manual*. OMG, 2003.

[7] A. Poniszewska-Maranda, G. Goncalves, and F. Hemery. Representation of extended rbac model using uml language. *Proceedings of SOFSEM 2005: Theory and Practice of Computer Science, LNCS 3381, Springer-Verlag*, 2005.

[8] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 1996.

[9] R. S. Sandhu and Q. Munawer. How to do discretionary access control using roles. *Proceeding of 3rd ACM Workshop on Role-Based Access Control, Fairfax*, 1998.

[10] R. S. Sandhu and P. Samarati. Authentication, access control and intrusion detection. *The Computer Science and Engineering hand-book*, 1997.

[11] J. B. Warmer and A. G. Kleppe. *The object constraint language. Precise modeling with UML*. Addison - Wesley, 1999.

Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)
0-7695-2362-5/05 $20.00 © 2005 IEEE