Abstract: - Access control administration regulates and enforces the definition of the components of an access control system by authorized administrative users. In this paper, the limitations on administrative aspects of well-known security models are addressed and the requirements for efficient management of authorizations are investigated in order to provide fine-grained and just-in-time access control for collaborative applications. Subsequently, the DARBAC (Dynamically Administering Role Based Access Control) model, which is a new approach in the area of role-based access control administration, is proposed and formally defined. The DARBAC model introduces new mechanisms for dynamic administration of access control and provides security features that comprise temporal role activation depending on particular objectives, controlled decentralization of administrative care, constraint-based privacy protection, dynamic separation of duties based on collaborative goals, and synchronization of permission availability for users with different responsibilities.

Key-Words: - Access control, meta access, RBAC, DARBAC

1 Introduction

Modern Web application systems introduce new trends in computing. However, determining the proper appropriate authorization schema in such conditions, results in a significant administrative overload [1]. In general, application security is characterized by a significant complexity due to the large number of variations and combinations of objects and operations to be protected [2]. This complexity increases in collaborative and distributed environments, as possible subjects may vary and their actions depend on contextual conditions that differentiate their need-to-know requirements. In such circumstances, access control administration tends to be costly and prone to error.

Access control ensures that accesses to the information system’s objects occur according to the modes and rules fixed by the corresponding security policy [3]. Subjects’ requests are permitted or denied usually by one (or more in the case of distributed systems) reference monitor that is responsible for mediating subject’s actions on objects according to a particular security policy, which is usually based on an access control model. Access control administration (or meta-access control [4]) regulates and enforces the definition of the components of an access control system by authorized administrative users. Access control administration workload depends on the number of handled components and on the frequency of changing permissions.

Mandatory or Multilevel Access Control (MAC) models [5], [6] restrict access to objects based on the sensitivity of the contained information and the clearance of subjects requesting to access such information. MAC administration is centralized; clearance of subjects is determined by the security administrator and sensitivity of objects is computed by the system based on the clearance of users.

Discretionary Access Control (DAC) models [7], [8], [9] restrict access to objects based on the identity of subjects. DAC administration is owner-based. However, Access Control Lists (ACLs) cannot be sufficient for the administration of access control in modern distributed and collaborative applications, where supporting dynamic changes of permissions, associating permissions to subject’s credentials when performing an operation, and relating permissions to content and attribute of resources or other contextual information [10], are critical requirements. Furthermore, the unlimited freedom of owners to delegate privileges often results in a complicated and uncontrolled administration of authorizations.

Role-Based Access Control (RBAC) [4], [11] has received considerable attention as a convenient way to simplify administration of access control by managing permissions through roles and their
hierarchies. Each user is assigned one or more roles, which are defined for various job functions, according to his qualifications and responsibilities [12]. Moreover, the use of role hierarchies provides additional advantages since one role may implicitly include the operations that are associated with another role. Hierarchies can represent the organization structure, which however is usually static, ordinary and vertical.

Access control in Web-based collaborative applications, must be based on permissions available (through activated roles) just-in-time to proper users accomplishing specific tasks. As discussed in section 2, current access control approaches present limitations to satisfactorily cope with such computing paradigms. Hence, new access control models to meet the requirements referred in section 3 are required. Moreover, new approaches in the area of access control administration, where often the security breaks [13], are also needed. An approach in this direction is the proposed DARBAC (Dynamically Administering Role Based Access Control) model that is presented in section 4. Next, the security features of the proposed model are discussed, followed by the concluding remarks.

2 Limitations of Current Approaches

Pure RBAC [12] seems to be suitable for function-oriented hierarchical organization structures usually used in relatively stable environments. Moreover, the use of roles for grouping authorizations is based on general organizational terms without incorporating user, object or process attributes [10]. Current organizational alternatives often adopt matrix organization or mission-oriented structures that lead to access control capable of embedding the required application-level contextual information. In addition, they introduce dynamic, process-based functional requirements for activity-intensive Web applications constituted by a complex mixture of tasks that may transcend organizational boundaries [1]. Therefore, application-level access control policies should be able to be expressed in new domains, like collaborative environments that allow users to cooperate on common tasks. Furthermore, in Internet-based collaborative environments, temporary task-force teams are created and rapidly changed. Team memberships are determined by both assigned roles and current functionalities of users. Task-force teams are dismissed when their goals are accomplished [14].

The three main approaches (MAC, DAC, RBAC) can not deal efficiently with such new requirements, since they support passive security models. Passive security models do not distinguish between permissions assignment and activation. In addition, they do not consider any contextual, objective or condition information to decide on a subject’s access request [15]. As a result, passive security models lack of fine-grained administration of authorizations. The specification of authorizations on the base of roles and object types seems to be insufficient for collaborative applications where fine-grained control on individual role and object instances is often the case. Even the mechanism of roles in RBAC is characterized as static, as opposed to the dynamic changes of context during the progress of collaborative activities [10]. Furthermore, the concept of session in RBAC restricts its applicability in large distributed and cooperative applications when many users are acting asynchronously but under conditions and constraints that are related to the same workflow or mission.

Administration of authorizations in RBAC is centralized. However, an approach to control distributing of administrative authority in a hierarchical way is offered in Administrative Role-Based Access Control (ARBAC) [16]. In addition, in the research community the increasing interest in just-in-time and active security, which is easy adaptable to particular environments, has lead to Task-Based Authorization Control (TBAC) and TeaM-Based Access Control (TMAC) models.

The TBAC model [13], [17] extends the traditional subject/object-based access control models by including domains that contain task-based contextual information. TBAC supports dynamic but centralized administration of authorizations, which are changed and granted in steps related to the progress of tasks.

The TMAC model [15] introduces the metaphor of teams, providing so a paradigm for access control that is natural to the collaborative work. Contextual information is associated with collaborative tasks and is further applied to decision making for activation of permissions. The collaboration context of a team contains the user context and the object context. At the time a team is instantiated, the user context is used to tailor the role-based permissions that are defined on object types to user-specific permissions on individual object instances, which are considered to be part of team's resources.

The Context-based TMAC (C-TMAC) model [18] integrates RBAC and TMAC concepts and uses a wider range of contextual information (time of access and location from which access is requested) that is incorporated in the overall architecture. C-TMAC preserves the advantages of scaleable security administration of RBAC model and yet offers more flexibility to control activation of
permissions for individual users and specific object instances. C-TMAC extends the notions of team and context, but it lacks the fine-grained and self-administration of entities and relations, as well as the definition of multidimensional contexts [10].

Finally, the PBDM family of models [19] also addresses the issue of authority delegation, dealing with various types of delegation: user-to-user, user-to-role, role-to-role. PBDM introduces however a great deal of administrative intervention for controlling or performing those delegations in large distributed application systems.

3 The need for Dynamic Access Control Administration

Access control administration seems to be basically simple as long as the sets of permissions do not change. However, information systems must reflect the changes of the organization they are serving. According to Sandhu [13], one of the main omissions of the RBAC model is the authorization of administration. Moreover, security models for Web applications must provide efficient and dynamic administration of authorizations to ensure that valid users exercise their privileges only during the progress of an official activity of their organization. Hence, additional access control mechanisms are needed to record dynamic changes in the content and context of information and monitor the state of the system [20]. The corresponding mechanisms must not overload the administrative task, by constituting a flexible and effective administration system. However, no proper concern on runtime administration of permissions and roles is given in known meta access models.

A number of important issues for applications in collaborative environments (recently discussed in the literature [10], [21]) include separation of duty, strict least privilege, order of events, delegation of authority, enforcement of access control at a distributed platform level, specification of permissions based on varied information like roles and context, scalability in terms of the quantity of collaborative operations, high granularity of information and resources protection, high level specification of permission, and dynamic specification and change of policies at runtime depending on the collaboration dynamics. Furthermore, a number of requirements for access control in workflow systems, as expressed in the form of regulations in [22], include the use of process instance-based user group, use of task as a factor for differentiating the access control mechanism, the privilege propagation function (from one role to another role in certain circumstances), the distinction between role and process instance-based user group, the ability for dynamic authorization based on access history, privacy protection through particular user exclusion, and temporal role delegation. Additional requirements that a workflow authorization model should support [23], include the need to grant authorizations to subjects only during the execution of a task and to revoke them immediately after the completion of the task, the ability to handle temporal constraints, and the use of role-based and event-based authorizations.

Another significant requirement for access control in collaborative environments is related to meta access control or access administration. Meta access control can either be incorporated within the basic access control model or provided through a separate model [2]. The DARBAC model, which is presented in the next section, can be considered as an access control model that incorporates meta access control for dynamic and distributed management of roles and permissions.

4 The DARBAC model

In a similar way to ERBAC [24], which introduces RBAC administration of authorizations in legacy ACL-based systems, the DARBAC model provides dynamic features that are met in workflow management [22], [23] and collaborative [15], [19] access control approaches. Moreover, the DARBAC model supports controlled access not only to data (objects) stored in a computer system, but also to processes used by subjects to manipulate those data. The DARBAC model relies upon two families of models, RBAC [12] and PBDM [19], and awards priority to the administration of access control during run-time. Specifically, a distributed Web-based information system, wherein a number of mission instances are in progress, is assumed. The protection state of the system is partitioned in particular states relating to each one mission instance. Mission instances can be conceived as loose variations of tasks that are performed by properly structured teams of users. An overview of the DARBAC model is shown in figure 1.

Fig. 1. The DARBAC model
4.1 DARBAC processes
The DARBAC processes are separated into two phases: build-time and run-time.

4.1.1 Build-time
Access control design is carried out in an offline state. The organization managers in cooperation with the application developers capture the organization's rules and policies to define, name and construct the components of the access control system: missions, objectives, organization roles, mission roles and their assignment relationships. In addition, particular permissions are defined and assigned to mission roles, which in turn are assigned to organization roles. The administrative operations during build-time are quite similar to those specified in the ARBAC family of models [16], [25], [26] that address sufficiently the distributed administration of role and permission assignments.

4.1.2 Run-time
Access control managing during run-time copes with situations related to any daily or emergency conditions in an organization. Administrative operations are performed by the security administrator, as well as regular users under certain constraints; e.g. a user can start a new mission instance and bind it to an objective instance. The security administrator creates users’ accounts and assigns them to organization roles. In order to perform a specific task, a user has to be accepted firstly for participation in a proper mission instance, according to the particular values of the objective instances bound to the mission instance.

4.2 DARBAC structure
The DARBAC model consists of six sets of entities called users, organization roles, mission roles, permissions, missions and objectives (Figure 1).

A user is a subject that uses a Web-application. Permissions are modes of access, which users can exercise on one or more data objects. Permissions are actually pairs, each one consisting of an object and an operation for that object. Roles used in DARBAC are distinguished between organization roles and mission roles. An organization role is a job function or title within the organization with some associated semantics regarding the authority and responsibility conferred on the user that is member of the organization role. Mission roles are application-specific and temporarily activated or delegated to other mission roles.

Mission roles are used as an intermediary between organization roles and permissions and are activated only in the frame of mission instances. Mission roles are grouped into organization roles that in turn are assigned to users. Furthermore, mission roles are characterized as regular (RR), fixed delegatable (FR), temporal delegatable (TR) and delegation (DR) roles, according to the PBDM2 model [19]. RR correspond to roles in the core RBAC [12], whereas FR, TR and DR are used for role-to-role delegation purposes. FRs own sets of DRs. TRs receive permissions delegated by FRs.

An important difference between DARBAC and other role-based approaches is that sessions are replaced by mission instances, which are more suitable for Web-application environments. A mission is defined as a type of project/process, that is carried out by a user or a team/group of users that are members of appropriate organization roles. In collaborative environments, missions are carried out by teams of users. A mission may also be entitled to a part (tasks) or an entire workflow. The structure of a mission represents a temporary project team organization, like an adhocracy organization [14]; for a finite period the members of a mission instance are acting in order to accomplish specific goals that are determined by a number of objectives. When goals are accomplished, the mission instance is ended.

The notion of objective is similar to context, as defined in [15], but in the DARBAC model it is used in a broader sense. An objective may also include the particular object of an operation; for example, the payment of a check concerns a given check that is identified by a serial number. In general, an objective instance may contain values of time intervals, object identities, and other contextual information that contributes to specify a restricted range to exercise generally applied permissions. An objective is defined by the Cartesian product of different sets of objective types, each one representing a domain of contextual information.

The following definitions provide some formalization to the above discussion.

Definition 1 – Entities:
- \( U, P, M \) stand for users, permissions and missions, respectively.
- \( O = \{2^{OT_1} \times 2^{OT_2} \times \ldots \times 2^{OT_n}\} \), an objective is a set of combinations of objective types \( OT_i \) from a finite set \( \{OT_1, OT_2, \ldots, OT_n\} \).
- \( O_i \), an objective instance is an n-tuple of values of an objective; for example the objective \( \{(21:30, 22:15), (Monday, Tuesday, Friday), (127.0.0.1)\} \).
that is defined for objective types time, day and IP address, respectively.

- **Mi**, is a mission instance.
- **OR**, is an organization role.
- **MR**, is a mission role that can be further refined in regular (RR), fixed delegatable (FR), temporal delegatable (TR) or delegation (DR) role.

Permissions are assigned to mission roles with a many-to-many Permission to Mission Role Assignment relation (PMRA). A mission role can have many permissions and the same permission can be assigned to many mission roles. Users are also related to organization roles with a many-to-many User to Organization Role Assignment relation (UORA). A user can own many organization roles, and an organization role can be assigned to many users. Furthermore, mission roles are assigned to organization roles with a many-to-many Organization role to Mission role Assignment relation (OMRA). Objectives are assigned to missions with a many-to-many Objective to Mission Assignment relation (OMA). The set of skills needed for a mission to accomplish its goal is specified through an Organization Role to Mission Assignment relation (ORMA), where each role refers to an autonomous activity of mission, giving an alternative way to represent tasks in workflows.

During run-time, users with sufficient administrative permissions can bind an objective to a mission instance through a many-to-one Objective instance to Mission instance Binding relation (OMB). In addition, when a user participates in a mission instance, a new entry is added in a many-to-many User to Mission instance Participation relation (UMP).

**Definition 2 – Relationships:**

- **PMRA** \(\subseteq P \times MR\), is a many-to-many permission to mission role assignment relation.
- **UORA** \(\subseteq U \times OR\), is a many-to-many user to organization role assignment relation.
- **OMRA** \(\subseteq OR \times MR\), is a many-to-many organization to mission role assignment relation.
- **UMP** \(\subseteq U \times M_i\), is a many-to-many user to mission instance participation relation.
- **OMB** \(\subseteq O_i \times M_i\), is a many-to-one objective instance to mission instance binding relation.
- **OMA** \(\subseteq O \times M\), is a many-to-many objective to mission assignment relation.
- **ORMA** \(\subseteq OR \times M\), is a many-to-many organization role to mission assignment relation.

Two types of constraints can be defined by security designers during build-time: Separation of Duty Constraints (SDC) and Join of Duty Constraints (JDC). SDC and JDC constraints are enforced during run-time to determine users’ participation in mission instances. SDC imposes the rule that no user can participate in a mission instance with more organization roles than the specified ones. JDC rules that a user, who is a member of a given organization role, can participate in a mission instance, subject to participation of a second user, who is a member of another organization role. In addition, identity-based inclusions/exclusions of users to participate in a mission instance can be specified by administrative or regular users during run-time with User-Mission Constraints (UMC); for example, a patient that controls accessing to his health-care record.

**Definition 3 – Constraints:**

SDC is expressed as a set of pairs \((rs, n)\). A set \(rs\) includes the organization roles, that are assigned to missions (as defined in ORMA) and also are in separation of duties. SDC imposes the rule that no one user can participate in a mission instance with \(n\) or more organization roles from the set \(rs\).

Let \(SDC_{m}\) be the set of pair \((rs, n)\) \(\subseteq (2^R \times N)\) that is defined for mission \(m \in M\), where \(rs \subseteq \{r \in OR | (r, m) \in ORMA\}\), \(n \in N\), \(|rs| \geq n \geq 2\). Let also \(sr_u\) be a subset of organization roles in \(rs\) that the same user may activate when participating in the same instance of mission \(m \in M\). Then, the cardinality of \(sr_u\) (the number of organization roles that the user can activate to participate in the same instance of mission \(m\)), cannot exceed the value of \(n\):

\[\forall (rs, n) : SDC_m, \forall sr_u \subseteq rs \bullet |sr_u| < n\]

JDC is expressed as a set of triples \((r_1, r_2, \oplus)\), where \(r_1, r_2 \in OR\) and \(\oplus\) is a symbol meaning that a user with organization role \(r_1\) can participate in a mission instance, only if another user with organization role \(r_2\) is already participating, regardless when the user with organization role \(r_1\) will stop participating in the same mission instance (\(\oplus\) is replaced by \(\triangleright\)),

- is not concurrently participating (\(\oplus\) becomes \(\#\)),
- is concurrently participating (\(\oplus\) becomes \(||\))

Let \(JDC_{m}\) be the set of triples defined for mission \(m \in M\). Let also \(DT_{r_i}\) be the period of participation of a user with role \(r_i \in OR\), \(i = 1, 2, \ldots N\), in an instance of mission \(m\). Then, for any instance of mission \(m\), each one replacement of symbol \(\oplus\) is defined as follows:

\[\forall (r_1, r_2, \triangleright) : JDC_m \bullet DT_{r_1} \cap DT_{r_2} \subseteq DT_{r_1}\]

\[\forall (r_1, r_2, \#) : JDC_m \bullet DT_{r_1} \cap DT_{r_2} = \emptyset\]

\[\forall (r_1, r_2, ||) : JDC_m \bullet DT_{r_1} \cap DT_{r_2} = DT_{r_2}\]
All users with appropriate organization roles can participate in an instance \( m_i \) of mission \( m \in M \) if constraints \( SDC_m \) and \( JDC_m \) are satisfied, unless specified differently in a constraint \( UMC_m \).

Let \( UMC_m \) be the set of pairs \((u, \otimes)\), specified for instance \( m_i \) of mission \( m \in M \), where \( u \in U \) and \( \otimes \) is a symbol. Then, a user \( u \) cannot participate in mission instance \( m_i \), if:

- an exclusion has been specified for that user (\( \otimes \) is replaced by symbol \(-\)), or
- inclusions have been specified for other users except user \( u \) (\( \otimes \) is replaced by symbol \(+\)).

5 The decision-making process
In the presence of a user access request, the DARBAC decision-making process is performed, based on the permissions the user acquires through mission roles and the particular values of the objective instances bound to the mission instance. The decision-making process is accomplished in a four-step procedure (as depicted in Figure 2):

Step 1: Reviewing
The user activates either manually or automatically (through the application) the proper organization role to participate in mission instance \( m_i \). In order to perform an operation, he uses the Web-application to submit the appropriate access request to the access control system, along with the name of the activated organization role. Then, given the bound objective instance, the mission instance \( m_i \) is sought in relation \( OMB \). In the case the mission instance \( m_i \) is not found, the access request is denied. Otherwise, relation \( UMP \) is sought for a previous participation of the user in the same mission instance. If such an entry is found, the procedure goes to step 3; otherwise it continues to step 2.

Step 2: Participating
The user’s participation in the mission instance is initially determined, according to the following sub-steps:

- Relation \( UORA \) is checked to confirm that the organization role presented by the user has been assigned to him. Otherwise, the user’s access request is denied and the procedure ends.
- Relation \( ORMA \) is sought for entries to permit a user to participate in an instance of mission \( m \), after he has activated the proper organization role.

Step 3: Checking
The dynamics associated with the current state of the access control system, as they are expressed by the constraints \( SDC_m \), \( JDC_m \) and \( UMC_m \), are examined to verify the user’s participation in the mission instance, as follows:

- The constraints \( SDC_m \), \( JDC_m \) and \( UMC_m \) are applied on the current values of names of user and organization role.
- In the case the above constraints are satisfied, the user is allowed to participate in the mission instance \( m_i \), (relation \( UMP \) is updated).

Besides the fact that the user already participates in the mission instance \( m_i \), this step has to be repeated for any new access request to verify his participation under the current dynamically changing conditions.

Step 4: Matching
The final judgment of whether the user has the right to execute his access request or not is taken according to the results of the following actions:

- Aggregation of user’s permissions, based on the mission roles either assigned to the organization role (regular roles) during build-time or delegated (delegation roles) during run-time, and
- Matching the requested operation to permissions.

6 Discussion
The DARBAC model provides a number of benefits that respond to the requirements presented in section 3. Decentralized and dynamic administration of authorizations is supported by granting to users the proper administrative permissions to manage mission instances and bind them to objective instances, to perform temporal delegations for controlled transfer of role competences, and to apply constraints governing mission participations.

Delegation and binding operations that are performed by a wide variety of users make the DARBAC-based access control to be self-administrative. Delegation is the ability of a user to distribute his privileges to others. Revocation undoes the effects of delegation. In general,
delegation is a helpful administrative tool for supporting decentralized authority or collaborative work. However, delegation is adopted by the DARBAC model in a totally temporary fashion, since its effect remains only during a mission instance and ends up automatically when a user with administrative permissions terminates the mission instance. This fact, in conjunction with the required participation of users in mission instances for being able to activate mission roles, ensures the automatic revocation of available permissions, either enriched via delegations or initially assigned ones.

Build-time constraints, JDC and SDC, provide rules governing the number and the sequence of users participating in mission instances. These constraints, along with mission instances, objective instances and role-to-role delegations, address the fundamental security issues of separation of duty, strict least privilege, and order of events [21]. Furthermore, run-time constraint UMC permits a user entitling the proper administrative permissions to control inclusion or exclusion of other users’ participation in a mission instance. The UMC constraint helps administration of authority not only to protect privacy but also to become fine-grained by specifying on a user basis the entities that are permitted to participate in a mission instance.

The concept of session is used in other RBAC-based access control models as a unity wherein the activation of roles and the enforcement of constraints are performed. Instead, in the DARBAC model mission instances are used in order to support enforcement of access control at a distributed platform level. A mission instance encompasses the whole effort of a team of users that aims to accomplish a goal through particular tasks. Hence, various constraints can be enforced without allowing users to violate them via multiple sessions. Mission instances are also used as a basis for controlling the activation of permissions assigned to mission roles owned by the same user. While sessions are user-oriented, missions are goal-oriented. As a result, the protection state can be checked on a collaborative (e.g. team) basis rather than on a single user basis.

Fine-grained and just-in-time access control is achieved using DARBAC administrative functions. Specifically, permissions are granted to users only during the execution of a task and revoked immediately after the task’s completion. Grants and revocations are performed in time, ensuring in this way a tight matching of permissions’ availability to actual need-to-do requirements. During build-time, mission roles can be assigned initially less permissions than needed. However, role-to-role delegation can be used in run-time to support further functional dependencies in the context of a mission instance; for example, to ensure that a particular task cannot be completed unless permitted by a supervisor role that delegates the necessary additional permissions through an appropriate delegation role [19].

Active security is provided through the use of contextual information. Besides location and time contexts, DARBAC’s objectives enclose additional information about conditions and objects of subjects’ access requests, as well as the purpose the mission serves. Hence, missions can be used either for simple user’s access requests (like the completion of a particular task) or complex groups of users’ access requests (as usually happens in collaborative environments).

The DARBAC model addresses the problem (as stated in [27]) of enforcing constraints at a behavioral level, such as least privilege, in order to eliminate vulnerabilities from insider threats. This is accomplished by allowing users first to activate one or more organization roles and then to ask for participation in specific mission instances, according to the access requests they submit through Web-applications (equivalent to selection of a task in the target system). Any set of permissions, which initially is assigned to a mission role, can be available only through the user’s participation in a proper mission instance. Furthermore, only the mission roles assigned to the prerequisite organization role through ORMA, and those gained through temporary role-to-role delegations, can be used to define the final set of permissions that the user can then exercise.

7 Conclusion

In this paper we proposed a new access control approach that preserves the advantages of permission administration that RBAC models offer. Moreover, the DARBAC model introduces the concept of mission, in addition to that of roles. Mission is used as an abstract mechanism to formulate the objective information, which in turn identifies the aim and context of activities to be performed by a group/team of users with prerequisite roles. Dynamic administration is achieved by granting to normal and administrative users the proper administrative permissions to manage missions and their objectives, to perform temporary delegations and to apply a number of constraints that concern the functional dependencies governing the participation of users in missions. In addition, a distinction is made between build-time
operations for static security analysis and design, and run-time operations for dynamic checks on the security aspects of the system to support a rich set of security policies that tune permission management in flexible ways. The proposed DARBAC model provides, amongst others, security features that comprise temporal role activation, controlled decentralization of administrative care, constraint-based privacy protection, dynamic separation of duties, and synchronization of permission availability for users with different responsibilities.

References: