Enhanced Sharing and Privacy in Distributed Information Sharing Environments

Ahmad Kamran Malik, Schahram Dustdar
Distributed Systems Group, Vienna University of Technology, Austria
{kamran, dustdar}@infosys.tuwien.ac.at

Abstract – **With the advancement in distributed computing and collaborative software technologies, information sharing and privacy related issues are gaining interest of researchers related to digital information creation, management, and distribution. Collaborative information sharing environment requires enhanced information sharing among users while privacy laws demand for the protection of user's information from unauthorized access and usage. Keeping this trade-off in view, there is a need for a flexible and enhanced information sharing model that preserves the privacy of user's information. We extend the Role-Based Access Control (RBAC) model to incorporate sharing and privacy related requirements and present a Dynamic Sharing and Privacy-aware Role-Based Access Control (DySP-RBAC) model. It is a family of models including core, hierarchical, and constrained RBAC models. The RBAC model is extended using team and task data elements as well as new data elements related to sharing and privacy of information. Sharing and privacy-based permission assignments and their conflict-handling strategies are described for a distributed and dynamic information sharing scenario.**

Keywords- Information Sharing, privacy, role-based access control, collaborative systems.

I. INTRODUCTION

With the advances in distributed computing and social software technologies [1], information sharing and privacy issues become critical. Distributed information sharing environments require enhanced information sharing among users [2] while privacy laws like HIPAA [3] demand for the protection of user's information from unauthorized access and usage. Keeping this trade-off in view, there is a need of an information sharing model that preserves the privacy of the user's information. Existing systems, based on access control technology, such as Role-Based Access Control (RBAC), are widely being used by enterprises to control the access to information and other resources. Conventional access control models do not handle privacy issues explicitly and cannot model enhanced sharing based on collaborative relationships among users. We extend RBAC to incorporate enhanced sharing and privacy requirements and present a Dynamic Sharing and Privacy-aware Role-Based Access Control (DySP-RBAC) model. The well known RBAC model which is based on [4] and standardized in [5], is extended to incorporate our sharing and privacy requirements. New entities like team, task, collaborative relationship, access level, purpose, condition, and obligation are introduced and their interactions are defined. Due to involvement of multiple data elements in our system, priority handling scheme is introduced based on smallest to largest data element priorities and exceptional priorities. In large and dynamic systems policy conflicts cannot be avoided. Conflict-handling mechanisms are introduced that use our priority handling mechanism. Methods and algorithms are provided for user's permission evaluation and conflict-handling.

The DySP-RBAC model is based on the notion of dynamic sharing and privacy of the information being shared among collaborating users working in multiple overlapping teams and tasks. A task and context-related information sharing among collaborating users helps in fulfilling collaborative tasks efficiently. In dynamic team and task-based environments, a user can join and leave her team whenever needed, and can participate in more than one teams and tasks at a time. Such a dynamic environment requires active access control, in which access decisions can be adapted at runtime using changing collaborative relationships and context conditions.

A privacy-aware system based on the RBAC model is described in [6] that extends RBAC model to include privacy data elements. It describes a privacy permission assignment language using purpose, condition, and obligation. Collaborative access control and privacy for Web-based social networks is described in [7]. A privacy policy model for enterprises is described in [8]. These systems concentrate only on privacy preserving without a focus on enhanced sharing among collaborating users. The DySP-RBAC model is an active access control model which is based on collaborative relationships and context of all involved entities in the system. In addition, it provides fine-grained access to information at individual user, team, and task level. Sharing requirements such as collaborative relationships, level of information sharing, and privacy requirements such as purpose, condition, and obligations are described in dynamic collaborative virtual teams scenario. The DySP-RBAC model is a family of models including core DySP-RBAC, hierarchical DySP-RBAC, and constrained DySP-RBAC.

The remainder of the paper is organized as follows. Section II describes dynamic sharing and privacy scenario. Section III explains the Dynamic Sharing and Privacy-aware Role-Based Access Control (DySP-RBAC) model. Section IV provides discussion about the rules and conflict-handling in DySP-

978-1-4577-2155-7/11/$26.00 ©2011 IEEE

RBAC model. Section V describes background and related work. Section VI concludes the paper and mentions future work.

II. Dynamic Sharing and Privacy Scenario

In distributed information sharing environment, individuals need to share their information with collaborating users for accomplishing their mutual tasks. Enterprises commonly use role-based access control to grant/restrict access to the information which is provided by collaborating users, while performing their tasks in teams. This information can contain certain level of user's personal or context information. Users are conscious about their information being used in a dynamic scenario where a user can frequently join and leave team and task. For such a dynamic scenario, we propose a dynamic sharing and privacy-aware model which is based on collaborative relationships among users. It exploits the fact that users who are in close relationship can access more information from others. We describe the types of relationships among users and their usage used in the DySP-RBAC model.

Our scenario uses five main data elements: enterprise, team, task, role, and user. Enterprise element is considered at the lowest priority level and user element is at the highest priority level. The enterprise element is not an explicit part of the DySP-RBAC model. Condition element (Con) is used to define enterprise-based conditions for collaborating employees of different enterprises. Priorities are based on collaborative relationships among users. For example, relationship between two users participating in the same task is stronger than the relationship between two users who are members of same team or same enterprise.

Collaborative relationships are the backbone of our system and are described as *Member*, *Mutual*, and *Colleague*. *Member* relationship is described as *Me* while *non-member* is described as *NMe*. Similarly *mutual* and *colleague* are described as *Mu* and *C* while *non-mutual* and *non-colleague* are described as *NMu* and *NC* respectively. *Member(Me)* relationship describes all those users who are working for same team. *Mutual(Mu)* relationship between two users describes that both users are performing the same task. Mutual is the closest in all relationships resulting in maximum number of collaborations within and across teams and enterprises. *Colleague(C)* relationship between two users describes that both users are employees of the same enterprise. Colleagues need not be a part of same team and tasks. There can exist complex relationships between collaborating users. For example there can exist a *member* relationship between two users while not being mutual and colleague.

III. Dynamic Sharing and Privacy-Aware Role-Based Access Control (DySP-RBAC) Model

DySP-RBAC model is based on NIST standard RBAC model [5] and is composed of three DySP-RBAC components: Core DySP-RBAC, Hierarchical DySP-RBAC and, Constrained DySP-RBAC. Hierarchical RBAC component of the RBAC model introduces role hierarchies where a role can inherit permissions from other roles. DySP-RBAC introduces team hierarchy, task hierarchy, CR hierarchy, Al hierarchy, Object hierarchy, and purpose hierarchy in addition to role hierarchies. Constrained DySP-RBAC contains static separation of duty and dynamic separation of duty components. Due to page limitations, we only describe core DySP-RBAC components in the following subsections.

A. Core DySP-RBAC

DySP-RBAC model shown in Figure 1.

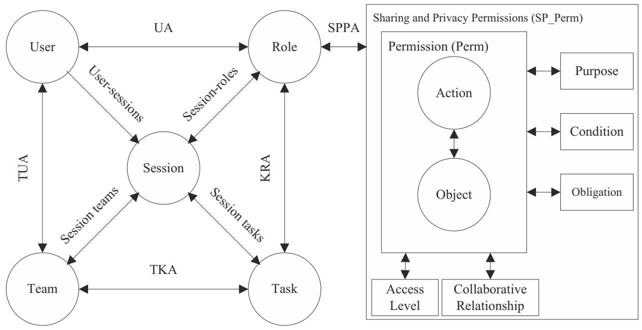

Fig. 1. Core DySP-RBAC model

The DySP-RBAC model extends the standard RBAC model. We introduce the notion of team and task data elements in addition to standard user, role, session, and permission data elements. For sharing and privacy, we introduce sharing elements Collaborative Relationships (CR) and Access Levels (AL), and privacy elements Purpose (Pur), context-based Condition (Con), and Obligations (Obl). User, in our system, is a human being. Role represents a job function in an enterprise. Objects, in our system, are information related to collaborating users, their team, task, resources, or their environment context. Action, in our system, is executable image of a program, which user can execute to perform some function. Permission (Perm) is an action allowed on an object. Sharing and privacy based permissions (SP_Perm) are the permissions that include sharing and privacy data elements. SP_Perm are assigned to roles. Conditions, in our system, include context-based privacy conditions as well as entity-based sharing conditions that allow or restrict one or more involved entities.

Data elements in our model that control the level of information sharing among collaborating users are CR and AL. CR element restricts the sharing of information to only those users who are in certain collaborative relationship with each other. Information, in our system, is related to context of a user, her environment, and tasks. This information is stored in different levels of hierarchies. AL element uses these information levels to control the sharing of information among users having different collaborative relationships and roles. Controlling information sharing using collaborative relationships can result in enhanced level of information flow among certain collaborating users, for example, users working in same team and performing same task will enjoy enhanced information sharing among each other to efficiently complete the mutual task. To preserve the privacy of user information that is going to be shared using CR and AL, DySP-RBAC uses Privacy elements Pur, Con, and Obl. Privacy elements restrict the information being shared, asking the purpose of information being accessed, and fulfilling the context-based conditions and certain obligation. Purpose of access should confirm to the purpose defined for the object being accessed, which is called purpose binding. Conditions related to user, her team, task, or environment must be satisfied. Obligations are requirements that need to be fulfilled to access the information, for example, logging the details of information shared with other users.

Two core elements introduced in DySP-RBAC to extend RBAC model are the team and task. One or more users are assigned to a team and one or more tasks are assigned to a team. Users being part of a team can participate in certain tasks that are assigned to them by the team. A user can participate in more than one team and more than one task at a time. Users in one task collaborate to perform the task and share their task related information as well as their context information among each other. Users in a team can be assigned same or different tasks created by their team. Being member of a team and performing team-tasks, a user can decide which level of granularity of information need to be shared with other collaborating users. Role is the most important data element in the RBAC model. DySP-RBAC extends the role of roles by assigning roles to tasks in addition to user-role assignment. One or more users can be assigned to one or more roles. Roles are also assigned to task. A user can participate in a task only if she is assigned the same role/s that is/are assigned to the task. A session is a mapping of one user to one or more roles, teams, and tasks. A user can establish a session and can activate her roles, teams, and task in that session. One session is associated to only one user and one user can establish more than one sessions at a time. For example, a user can establish more than one session for different teams or she can activate more than one teams and tasks in one session.

B. Formal Description of Core DySP-RBAC

Here we describe the data elements, their assignments, and functions used in Core DySP-RBAC model. Following are the data elements in DySP-RBAC model:

U, R, Obs, Ops, T, K, Pur, Con, Obl, CR, AL are users, roles, objects, operations/actions, teams, tasks, purposes, conditions, obligations, collaborative relationships, and access levels.

Following are the RBAC model data elements that are also used in DySP-RBAC:

- U: the set of users in our model.
- R: the set of roles.
- 2^R: the power set of R.
- Obs: the set of objects that need to be accessed/shared.
- Ops: the set of operations on objects. Operations are executable image of a program that performs some operation on objects when invoked by user. An operation is described as ”action” in DySP-RBAC model.

Following are the main data elements added in DySP-RBAC model to extend RBAC model:

- T: the set of teams in the system.
- K: the set of tasks assigned to a team.

Following elements are added to preserve the privacy of user's information:

- Pur: the set of purposes defined for accessing objects in the system.
- Con: consists of disjunction of conjunctive statements containing context variables, operators, and values.
- Obl: the set of obligations that must be fulfilled as a result of access permission.

Following elements are added to control/enhance sharing of information:

- CR: the set of collaborative relationships defined in system.
- AL: the set of access levels (hierarchical levels defined for objects) defined to allow access at certain level of granularity.

Following are the assignment relations among elements of DySP-RBAC model:

- $UA \subseteq U \times R$, a many to many mapping user-to-role assignment relation.
- $Perm = 2^{(Ops \times Obs)}$, the set of permissions.
- $SP\_Perm = (Perm, CR, AL, Pur, Con, Obl)$, the set of sharing and privacy based permissions.
- $SPPA \subseteq SP\_Perm \times R$, a many to many mapping sharing and privacy based permission-to-role assignment relation.
- $KRA \subseteq K \times R$, a many to many mapping task-to-role assignment relation.
- $TKA \subseteq T \times K$, a many to many mapping task-to-team assignment relation.
- $TUA \subseteq T \times U$, a many to many mapping user-to-team assignment relation.

Following are the function mappings used in DySP-RBAC model:

- $assigned\_users : (r : R) \rightarrow 2^U$, the mapping of role r onto a set of users.
- $assigned\_users(r) = u \in U \mid (u, r) \in UA$.
- $assigned\_permissions : (r : R) \rightarrow 2^{SP\_Perm}$, the mapping of role r onto a set of SP-based permissions.
- $assigned\_permissions(r) = \{p \in SP\_Perm \mid (p, r) \in SPPA\}$.
- $assigned\_team\_users : (t : T) \rightarrow 2^U$, the mapping of team t onto a set of users.
- $assigned\_team\_users(t) = \{u \in U \mid (u, t) \in TUA\}$.
- $assigned\_team\_tasks : (t : T) \rightarrow 2^K$, the mapping of team t onto a set of tasks.
- $assigned\_team\_tasks(t) = \{k \in K \mid (k, t) \in TKA\}$.
- $assigned\_task\_roles : (k : K) \rightarrow 2^R$, the mapping of task k onto a set of roles.
- $assigned\_task\_roles(k) = \{r \in R \mid (r, k) \in KRA\}$.
- $assigned\_user\_tasks : (u : U) \rightarrow 2^K$, the mapping of user u onto a set of tasks.
- $assigned\_user\_tasks(u) = \{k \in K, t \in T \mid (t, u) \in TUA \wedge (t, k) \in TKA\}$.

Following are the details of sessions and their mappings:

- $Session\ S$: the set of sessions.
- $user\_sessions(u : U) \rightarrow 2^S$, the mapping of user u onto a set of sessions.
- $session\_user(s : S) \rightarrow u \in U$, the mapping of each session s_i to a single user of the session s_i.
- $session\_roles(s : S) \rightarrow 2^R$, the mapping of session s_i onto a set of roles.
- $session\_roles(s_i) \subseteq \{r \in R \mid (session\_user(s_i), r) \in UA\}$.
- $session\_teams(s : S) \rightarrow 2^T$, the mapping of session s_i onto a set of teams.
- $session\_teams(s_i) \subseteq \{t \in T \mid (session\_user(s_i), t) \in TUA\}$.
- $session\_tasks(s : S) \rightarrow 2^K$, the mapping of session s_i onto a set of tasks.
- $session\_tasks(s_i) \subseteq \{k \in K, t \in T \mid (session\_user(s_i), t) \in TUA \wedge (t, k) \in TKA\}$.
- $session\_team\_tasks(s : S, t : T) \rightarrow 2^K$, the mapping of team t onto a set of tasks in a session.
- $session\_team\_tasks(s_i, t) \subseteq \{k \in K \mid (t \in session\_teams(s_i)) \wedge (k \in session\_tasks(s_i)) \wedge (t, k) \in TKA\}$.
- $session\_task\_roles(s : S, k : K) \rightarrow 2^R$, the mapping of task k onto a set of roles in a session.
- $session\_task\_roles(s_i, k) \subseteq \{r \in R \mid (r \in session\_roles(s_i)) \wedge (k \in session\_tasks(s_i)) \wedge (r, k) \in KRA\}$.
- $session\_team\_roles(s : S, t : T) \rightarrow 2^R$, the mapping of team t onto a set of roles in a session.
- $session\_team\_roles(s_i, t) \subseteq \{r \in R \mid \bigcup_{k \in session\_team\_tasks(s_i,t)} \{session\_task\_roles(s_i, k)\} \wedge session\_roles(s_i)\}$.
- $session\_team\_permissions(s : S, t : T) \rightarrow 2^{SP\_Perm}$, the SP permissions available to a team t in a session, $\bigcup_{r \in session\_team\_roles(s_i,t)} \{assigned\_permissions(r)\}$.
- $avail\_session\_permissions(s : S) \rightarrow 2^{SP\_Perm}$, the SP permissions available to a user u in a session, $\bigcup_{r \in session\_roles(s_i)} \{assigned\_permissions(r)\} \cup \bigcup_{t \in session\_teams(s_i)} \{session\_team\_permissions(s_i, t)\}$.

C. User Permissions in $User\_Sessions$

A user, in the DySP-RBAC model establishes a session and activates some roles called $session\_roles$ and teams called $session\_teams$. After that, only those tasks can be activated by the user that are assigned to the user by a $session\_team$ and the roles required for those tasks ($assigned\_task\_roles$) are included in user's active roles ($session\_roles$). Activated tasks are called $session\_tasks$. Tasks activated in a $session\_team$ are called $session\_team\_tasks$. Roles activated in a $session\_task$ are called $session\_task\_roles$ and roles activated in a $session\_team$ are called $session\_team\_roles$ which are the roles required by all $session\_team\_tasks$. $session\_team\_roles$ are the intersection of $session\_task\_roles$ and $session\_roles$. Permissions available to a $session\_team$ called $session\_team\_permissions$ are the union of permissions in all $session\_team\_roles$. Permission available to a $session\_user$ are the union of permissions in all $session\_roles$ and $session\_team\_permissions$.

IV. Discussion: DySP-RBAC Rules and Conflict Handling

In this section, we describe permission assignments, prohibitions, and exceptional rules used in DySP-RBAC model. These rules are called sharing and privacy-aware permission assignment (SPPA), sharing and privacy prohibitions (SP_Proh), and sharing and privacy exceptions (SP_Except) respectively. In presence of many data elements and their conflict of interests, conflicts in sharing and privacy rules cannot be avoided. A conflict handling algorithm for conflicting rules is provided which uses priorities of data elements.

A. Permissions Assignments, Prohibitions, and Exceptions

Different types of data elements are used in the DySP-RBAC model including RBAC-based data elements, sharing-aware data elements, privacy-aware data elements, and two new data element assigned to user called team and task. Conditions, in the DySP-RBAC model, can be described as simple or complex conditions. Simple conditions consist of a condition variable, an operator, and condition value. Simple conditions can be defined as: $(cond\_var\ op\ val)$. Complex conditions consist of disjunctions of conjunctive statement.

Condition variables in the DySP-RBAC model are either context variables or are data element-based conditions which include team, task, user, and enterprise. Data element based conditions provides instance level permissions and result in providing fine-grained access control. To control the level of information sharing and privacy, data elements and conditions are used in three types of rules. Sharing and privacy permission assignments, prohibitions, and exceptional rules are formally defined below.

- Sharing and Privacy-aware Permissions Assignments ($SPPA$).
  - Data permission $Perm$ consist of object Obs and their operations Ops. The set of data permissions is defined as:

    $Perm = \{(obs, ops) \mid obs \in Obs, ops \in Ops\}$.
  - Sharing and privacy-aware permissions SPP contain sharing and privacy data element and are defined as:

    $SPP = \{(perm, pur, con, obl, cr, al) \mid perm \in Perm,$
    $pur \in Pur, con \in Con, obl \in Obl, cr \in CR,$
    $and\ al \in AL\}$.
  - Sharing and Privacy-aware Permissions Assignment ($SPPA$) is defined as:

    $SPPA \subseteq R \times SPP$.

$SPPA = \{(r, perm, pur, con, obl, cr, al) \mid r \in R,$
$perm \in Perm, pur \in Pur, con \in Con, obl \in Obl,$
$cr \in CR,$and$al \in AL\}.$

- Sharing and Privacy-aware Prohibition (SP_Proh) in the DySP-RBAC model is the rule to restrict access of one user or more than one user (being member of certain data element like role, team, task etc.) to certain object under certain conditions and is defined like an SPPA as:

  $SP\_Proh = \{(r, obs, ops, pur, con, cr, al) \mid r \in R,$
  $obs \in Obs, ops \in Ops, pur \in Pur, con \in Con,$
  $cr \in CR,$and$al \in AL\}.$

- Sharing and Privacy-aware Exception (SP_Except) in the DySP-RBAC model is the highest priority rules to allow/restrict access of one user or more than one user (being member of certain data element like role, team, task etc.) to certain object under certain conditions and is defined as:

  $SP\_Except = \{(r, obs, ops, pur, con, cr, al, decision) \mid$
  $r \in R, obs \in Obs, ops \in Ops, pur \in Pur, con \in Con,$
  $cr \in CR, al \in AL, anddecision \in (+, -)\}.$

Sharing and privacy-aware rules are based on roles. As roles are also assigned to tasks in the DySP-RBAC model, so at runtime a user can activate only those $assiged\_roles$ that are also assigned to her tasks. A permission for a specific user, team and task can also be defined in the DySP-RBAC model and such condition is specified using the condition data element ($Cond$). In presence of hierarchies, many permission assignments (SPPA) containing same action can be replaced by one SPPA. This is due to the fact that a higher level data object in a hierarchy inherits permissions of all its lower level data objects.

B. Examples of SPPA, Prohibition, and Exceptional rules

The DySP-RBAC model uses three types of rules: Sharing and Privacy-aware Permission Assignments (SPPA), Sharing and Privacy-aware Prohibitions (SP_Proh), and Sharing and Privacy-aware Exceptions (SP_Except). SPPA, also known as positive rules, are used to allow access to one or more than one user (having certain data elements like role, team, task etc.) at certain level of information under certain conditions. An example of SPPA is described here:

$SPPA = (Proj\_Mgr, location, read, management, \phi, \phi,$
$Me, L2)$

This example describes that owner of the context object $location$ has defined an SPPA rule to allow her Project Manager to know her current location details for management purpose up to the level (L2) of detail only if she is member of the same team as owner (Me describes member relationship).

Prohibition (SP_Proh), also known as negative rules, are used to restrict access of one or more than one user (having certain data elements like role, team, task etc.) at certain level of information under certain conditions. An example of SP_Proh rule is described here:

$SP\_Proh = (App\_Dev, online\_status, read, \phi, \phi, \phi,$
$NMu, \phi)$

This example describes that those application developers cannot read online status of the owner who are not taking part in same task as the owner (NMu describes non-mutual relationship which means two users are not working for the same task). These rules can also be applied to a group of users independent of role, for example, users working in same team, task, or enterprise. Another example of SP_Proh rule is described here:

$SP\_Proh = (\phi, accessible\_device, read, \phi, team = t1, \phi,$
$\phi, \phi)$

It describes that all users of team $t1$ are prohibited to read about the accessible devices of the owner. If the owner of information wants to allow some users of team $t1$ to access this information, it is impossible using an SPPA rule because prohibitions take precedence in the DySP-RBAC model. To handle this issue, we need exceptional priority rules (SP_Except). Exceptional priority rules (SP_Except), are highest priority rules used to allow/restrict access of one user or more than one user (having certain data elements like role, team, task etc.) at certain level of information under certain conditions. An example of SP_Except rule is described here:

$SP\_Except = (\phi, accessible\_device, read, \phi, team = t1 \wedge$
$task = k1, \phi, \phi, \phi, +)$

This exceptional rule allows to read about the accessible devices of the owner, to only those members of team $t1$ who are also taking part in task $k1$. The plus sign here is used to allow access while a minus sign is used for deny access.

C. Conflict Handling and Priority

Conflicts can occur in sharing and privacy rules of the DySP-RBAC model either due to conflict of interest of different data elements, or due to conflicting conditions, obligations, and access levels. Conflicts in two or more obligations or two or more access levels are prominent and easy to find by comparing their values. While conflicts in conditions can be tricky and need comparisons of condition variables and their values. In this work, we are interested to describe the conflict handling strategy for conflicting data elements because the DySP-RBAC model is related to information sharing and privacy based on collaborative relationships of data elements.

Conflicts among rules often occur in presence of many data elements and their different priorities. We describe methods that define priorities for data elements and handle rule conflicts. Each data element is assigned a priority level that helps to decides rule conflicts among conflicting elements. Positive rules are used to allow access while negative rules are used to deny access to a data object. In some systems where both positive and negative rules are used, either positive or negative rules are given priority over the other to handle conflict among rules. There can be some requirements that cannot be handled by group-based and rule-based priorities, these special cases are handled by defining exceptional priority rules. Priority and conflict handling methods are shown in Algorithm 1.

Algorithm 1 Priority-based Conflict Handling Algorithm

```
[Find SPPA, Proh, and Except rules for query evaluation]
if (found $an$ Except $rule$) or ($found$ an $SPPA$ or $Proh$ without $a$ conflict) then
  Apply the rule
else if found $conflicting$ rules then
  [Find and Compare Elements in rules]
  Apply rule containing smaller element in CR or Con
  if found $same$ level $elements$ in rules then
    Apply Proh rule
  end if
else if no $rule$ found then
  Apply default closed policy
end if
```

It describes conflict handling strategy for data elements in presence of positive and negative, sharing and privacy rules. In case of a conflict detection, first it uses exceptional priority (if exists), after that, it compares data elements and smaller data element gets priority. In presence of a conflict caused by same data element, prohibition gets priority over authorization.

V. BACKGROUND AND RELATED WORK

Access control researchers have extended the RBAC model for their specific requirements. For example, Team-based Access Control (TMAC) model [9] and Task-Based Access Control (TBAC) model [10] extend the RBAC model to assign permissions based on team and task of the user. They do not handle privacy and enhanced sharing requirements and collaborative relationships among users. A number of languages and systems have been described in literature to enforce privacy by extending the RBAC model. For example, a privacy-aware system using the RBAC model is described in [6]. It provides a privacy permission assignment language using purpose, condition, and obligation. Another privacy policy model for enterprises is described in [8]. These systems concentrate only on privacy preserving without a focus on enhancing sharing among collaborators. Our owner-defined roles for context sharing are described in [11].

Access control policy for collaborative environments and their requirements have been described in [12]. The use of RBAC model in collaborative systems is explained in [13]. The RBAC model is an efficient model for management of permissions in large-scale systems. Still it lacks in fulfilling the privacy, enhanced sharing, and fine-grained access level requirements of collaborative systems.

Authorization, obligation, and their conflicts resolving strategies are described in [14]. Static and dynamic conflict are described and their detection methods are provided in [15]. An owner-based hybrid policy-based sharing control model is described in our system [16]. The DySP-RBAC model uses priorities of data elements and exceptional priorities to resolve conflicts occurring in sharing and privacy-aware permission assignment and prohibition rules.

VI. CONCLUSION AND FUTURE WORK

This paper describes sharing and privacy of user's information in a collaborative working scenario and explains our DySP-RBAC model which extends the RBAC NIST standard model. We include team and task data elements and a number of data element related to sharing and privacy including purpose, condition, obligation, collaborative relationship, and access level. The DySP-RBAC model is a family of models including core DySP-RBAC model, hierarchical DySP-RBAC model, and constrained DySP-RBAC model. Examples of sharing and privacy-aware permission assignments, prohibitions, exceptions, and their conflict handling strategies using priorities of data elements are discussed. In future, we would like to dig into the details of different types of permission assignments, context conditions, and conflict-handling in different scenarios.

REFERENCES

[1] M. N. Kamel Boulos and S. Wheeler, "The emerging web 2.0 social software: an enabling suite of sociable technologies in health and health care education," *Health Information & Libraries Journal*, vol. 24, no. 1, pp. 2–23, 2007.

[2] K. Smith, L. Seligman, and V. Swarup, "Everybody share: The challenge of data-sharing systems," *IEEE Computer*, vol. 41, no. 9, pp. 54–61, 2008.

[3] "United state department of health. Health insurance portability and accountability act of 1996," Available at http://www.hhs.gov/ocr/hipaa/.

[4] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.

[5] D. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," ACM Trans. on Information and System Security (TISSEC), 4(3): 224-274, Aug. 2001.

[6] Q. Ni, E. Bertino, J. Lobo, C. Brodie, C.-M. Karat, J. Karat, and A. Trombeta, "Privacy-aware role-based access control," *ACM Trans. Inf. Syst. Secur.*, vol. 13, pp. 24:1–24:31, July 2010.

[7] B. Carminati and E. Ferrari, "Access control and privacy in web-based social networks," *IJWIS*, vol. 4, no. 4, pp. 395–415, 2008.

[8] G. Karjoth and M. Schunter, "A privacy policy model for enterprises," in *Proceedings. 15th IEEE Computer Security Foundations Workshop, 2002*, 2002, pp. 271 – 281.

[9] C. K. Georgiadis, I. Mavridis, G. Pangalos, and R. K. Thomas, "Flexible team-based access control using contexts," in *Proceedings of the sixth ACM symposium on Access control models and technologies, SACMAT '01*, New York, NY, USA, 2001, pp. 21–27.

[10] R. Thomas and R. Sandhu, "Task-based authorization controls(TBAC): Models for active and enterprize-oriented management," Database Security XI, Holland, 1997.

[11] A. K. Malik and S. Dustdar, "Context-aware sharing control using hybrid roles in inter-enterprise collaboration," ICSOFT 2010, Athens, Greece, 22-24 July, 2010.

[12] H. Shen and P. Dewan, "Access control for collaborative environments," in *Proceedings of ACM CSCW'92 Conference on Computer-Supported Cooperative Work*, 1992, pp. 51–58.

[13] G.-J. Ahn, L. Zhang, D. Shin, and B. Chu, "Authorization management for role-based collaboration," in *IEEE International Conference on Systems, Man and Cybernetics, 2003.*, vol. 5, oct. 2003, pp. 4128 – 4134.

[14] E. Lupu and M. Sloman, "Conflicts in policy-based distributed systems management," *IEEE Transactions on Software Engineering*, vol. 25, no. 6, pp. 852–869, Nov/Dec 1999.

[15] N. Dunlop, J. Indulska, and K. Raymond, "Dynamic conflict detection in policy-based management systems," in *Proceedings. Sixth International Enterprise Distributed Object Computing Conference, EDOC '02*, 2002, pp. 15–26.

[16] A. K. Malik and S. Dustdar, "A hybrid sharing control model for context sharing and privacy in collaborative systems." AINA 2011, Singapore: IEEE Computer Society, 2011, pp. 879–884.