# *Privacy based temporal rbac model*

S.SATHEA SREE
Department of CSE, MNM Jain Engineering College, Chennai.
satheasadasivam@gmail.com

**Abstract—**Privacy has been acknowledged to be a critical requirement for all organizations. Specifying such kind of privacy policy based on role based access control model is crucial. With ubiquitous computing technologies, business processes become more mobile and distributed and are executed in various objects. Context-aware access control mechanisms are an important perquisite to protect sensitive data and services in secure ubiquitous computing environments. A context constraint specifies that certain condition that must be fulfilled to permit the execution of a particular task. Conflict resolution is one of the main issues in permission assignment in the Privacy Aware Role Based Access Control Model. There could be a possibility that there is no conflict for up to two permission assignments, but there is a chance of getting conflict when three or more PA's are considered together.

To overcome this issue we aim to extend the conflict detection algorithms which will detect the conflicts for up to N permission assignment's and we integrate context constraints with process-related role – based access control(RBAC) models and thereby support context-dependent task execution to improve the efficiency of the User Assignment we incorporate Rule design to assigning users to roles

***Keywords***- Splitting Context Variable; Permission Assignment; Privacy aware RBAC and Temporal Constraints

## I. INTRODUCTION

Nowadays the privacy plays the vital role in deciding the security over the informations in the system. Privacy policies are acting as the access control

rules to protect the system from the unauthorized access. The security provided by the traditional access control models is not adequate for the upcoming requirements of the latest technology. After that the role based access control models have been introduced. This also does not satisfy the privacy requirements. So there is a strong need to have an efficient system
that should define the strict privacy policies in a way that should not be breakable by any one.

In order to come up with best efficiency and up to date issues, most organizations create some roles and assign set of permissions to that role. Breaking the privacy on sensitive informations is considered as violation of rules. Therefore, it is very much required that the permissions to the roles have to be given based on temporal-period .

## II. ROLE BASED ACCESS CONTROL (RBAC)

In the past years the role-based access control (RBAC) has been established for security administration needs, and it received strong support from the research and practitioner communities. In this short duration it has become the best form of access control mechanisms in business enterprises.

In every computer, access control is an approach for restricting system access to authorized users. Role based access control is a one way in which the access control is practiced through roles. Inside an organization, roles are established to mean various job functions. The permissions are assigned to specific roles to perform some actions. Members of team (or other system users) are assigned particular roles, and through those role assignments acquire the subset of the permissions assigned to roles, to perform particular system functions. Since users are not assigned permissions directly, they acquire them through their role (or roles), management of individual user rights becomes a matter of simply assigning appropriate roles to the user; this simplifies the operations, such as adding a user, or changing a user's department. It includes the Components as follows:

1. Role assignment: A subject can execute a function only if the subject has been assigned a role previously.
2. Role authorization: A subject's active role must be authorized for the subject. This ensures that users can acquire on only roles to which they are authorized.
3. Transaction authorization: A subject can execute a transaction only if the transaction is assigned for the subject's active role. This ensures that users can execute only transactions for which they are authorized.

### *A. PRIVACY AWARE ROLE BASED ACCESS CONTROL*

Traditional access models, such as Mandatory Access Control (MAC) and Discretionary Access Control (DAC), are not designed to enforce privacy policies rather barely meet the requirements of privacy protection. However, existing access control technology can be used as a starting point for managing personal identifiable information in a trustworthy fashion.

The Role Based Access Control model is an alternative to the conventional access control models. This includes three models such as, core-RBAC, Constrained RBAC, Hierarchy-RBAC. The latter two models are extended from the core model.

In order to extend classical RBAC to support complex privacy policies, consistently with the approach adopted for classical RBAC, we take the approach of defining a family of Privacy aware RBAC (P-RBAC) [2] conceptual models characterized by different modeling capabilities.

***A. Core P-RBAC***: is the base model, placed at bottom. It satisfies all the fundamental features of the RBAC model. It should have sufficient expressive power for representing public privacy policies, privacy statements and privacy notices in Web sites, and policies based on privacy related acts. On the other hand, conflicts detection in Core P-RBAC should remain tractable. Advanced models in the family extend Core P-RBAC with additional modeling constructs.

***b.Hierarchical P-RBAC*:** introduces the notions of *Role Hierarchy* (RH), *Data Hierarchy* (DH), and *Purpose Hierarchy* (PH); it thus enhances Core P-RBAC with hierarchical organizations for three important entities of Core P-RBAC.

***c.Conditional P-RBAC***: It provides common constructs for building the components of *core P-RBAC*. Also introduces Permission Assignment Sets and Boolean Expressions; its main goal is to provide a language for expressing conditions richer than the simple condition language provided by Core P-RBAC. Universal P-RBAC combines functionalities of both Conditional P-RBAC and Hierarchical P-RBAC.

The three main components of P-RBAC are purpose binding, conditions and obligations.
Purpose binding**:** It means that data collected for one purpose should not used for another purpose without user consent.
Conditions: They are the prerequisites to be met before any action can be executed or permission can be assigned.
Obligations: They are the actions to be performed after a permission has been assigned and some action is executed on data objects to make the action complete.

## 1. ACTIVITY CONSTRAINT MODEL

Pervasive Computing integrates the physical environment space with the user space. It allows users to interact with the environment in a way that allows users to reduce their focus on computing technology and concentrate more on their current tasks. Designing a pervasive system requires integration of all areas of computer science and engineering from hardware designs to theoretical studies.

The area of Pervasive Computing which this research addresses is access control. Though the system would eventually be designed and implemented, security and trust issues could prevent it from being used. Users will interact with the smart environment with interactive applications on peripheral devices that communicate with the system providing services to their current task.

The environment contains applications or brokers waiting for requests of service to carry out tasks of which it was designed. Access control to various objects, files, or devices becomes necessary to the success of Marc Weiser's vision [69]. This vision focuses on the seamless interaction between users and the environment filled with embedded computing systems. These objects could be any household or office appliance, electronic files, or peripheral devices. In general, the objects requiring protection are physical devices or virtual files.

Pervasive Computing brings other issues that raise concern because of its high mobility and service capabilities. One could request a service that wishes access to a resource without regard for activities occurring in the environment. Such a request could

violate the user intentions of the current activity. Actions by users in a space continuously
affect the security properties of a smart environment where they cannot always see or hear
all actions or events occurring. Thus, Pervasive Computing brings the issue of preserving user intents in a physical environment where users are consistently interacting with the space while still applying the appropriate security policies and preventing unauthorized accesses.

## 2. ROLE-CARDINALITY ACTIVATION CONSTRAINT

Conflict of interest in ABAC may occur when a user engages with other users where their skills or interests are divergent. This does not mean that users may not be associated with other users of different skill code or interest. This means the information shared within an activity is restricted only to authorized users, not necessarily requiring users to be associated with the same role. Separation of duty on activated role in an activity is to enforce the constraint on the assignment of users to activities. Such a constraint would require an activity, such as a parent-teacher meeting, to have a minimum of 1 parent and 1 teacher role activated before the activity could activate.

## 3. ACTIVITY SEPARATION

Within a given environment or setting, conflicts of interest occur when activities are occurring simultaneously or too many activities of the same type are concurrently occurring. Mutually exclusive activities (MEA) allows activities to be authorized when two or more activities do not create a conflict of interest when acted on independently, but produce policy concerns when activated simultaneously.

## 4. CONTEXT CONSTRAINT

We provide two types of context constraints: activity context constraint and role context constraint. Context conditions are applied to both activities and roles associated with the activity. For an activity or role to be active, all context conditions associated with the entity must be true. Associated with each condition is a set a context variable that must be active and ready to be validated. The Activity-based Access Control Model context constraint validation only occurs if the role requirements, role-cardinality activation constraint, and activity separation have been validated and satisfied. Upon a session's association with
an activity, only then are the associated contexts variables subscribed to and received from the context provider. Hence, an active view of all contexts is not required, but only relevant context to the activity is supplied. Users requesting to take part in an activity are allowed to utilize the permissions only when all validations have been satisfied.

Throughout the duration of an activity, revocations of activities and roles may occur based on environment and user context. A generalization required in the context constraint language is that any constraint may not be directed towards a specific user. Since we are taking advantage of RBAC's concept of roles, we must retain that advantage. Besides, specifying constraints directed towards each user is inefficient and difficult to maintain.

Instead, constraints are directed towards the roles associated with an activity. In turn, users associated with the role and activity will be associated with the constraint. Violations of activity context constraints will deactivate the activity, and violations of role context constraints will deactivate those roles associated with the activity. Notifications and time-outs are required before deactivation to avoid abrupt actions from occurring
unless the violation is critical.

## 5. ACTIVITY

To authorize a user the system must determine the request source. Since Pervasive Computing environments are information-rich in context, this information is used to verify the environment conditions of the participating users and brokers to ensure the purpose of the activity. Through Activity-based Access Control Model context constraints, activities are limited to the context conditions specified in a policy.

## 6. ROLES

As well as activity constraints, we separate context constraints for activities as a whole and each individual participant. Activity context constraints deactivate an activity if any of its constraints are not satisfied, while activity role constraints deactivate only subject roles. This prevents participation of only specified role when those constraints are not satisfied. This avoids deactivation of an activity when it is the case that a single role does not satisfy the constraints; and without the active role the activity still satisfies the constraints.

# III. PROCESS –RELATED RBAC

One objective of our research is to define process-related context constraints via native modeling

language constructs. Usually, process models focus on the process-flow perspective and are decoupled from access control-relevant context information.

*A.* ***Access Control Layer:***

This Layer consists of three major working modules such as user assignment, permission assignment, and conflict detection engine. All the techniques are used to build a flexible role structure system.

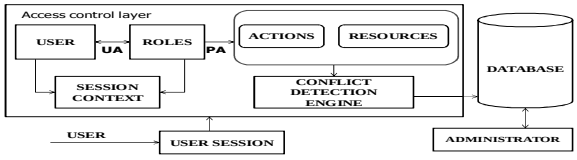


Fig. 3.1 System Architecture

***B.Representing Time:***

*In o*ur model we have used two kinds of temporal constraints. They are instant time and interval time instant time specifies a specific time like 10 a.m. The interval time means duration of some time like 7 p. m to 9 p.m.

The context variable to store the current time is not a splitting context variable; it won't split the data based on the value it assumes. The conflict in permission assignment can be reduced by using the time constraint in defining privacy policies. The conflict detection algorithm detects the conflicts by comparing the values of the context variables stored in the scope list as follows

PA3*: (GMGR, ((RD, CHA), TOTALDEPOSIT, BB =Chennai Λ BC=yes Λ BL= Perungudi*, Λ CT=10 AM-12 AM, *notify by (email)))*

Here the General Manager can read the Chennai account for the purpose of calculating the total deposit when the bank branch is Chennai, branch consent=yes, Bank location is perungudi and current time is10 A.M to 12 A.M however, Chennai branch will be informed by official Email."

***C.Conflict Detection Module:***

The conflict detection engine (CDM) gets the help from RBAC manager to compare newly arrived request for permission assignment with the existing defined policies. The RBAC manager gets the list of attribute from the database and assist conflict detection engine. CDM verifies the fuzzy condition to check whether conflict occurs on given request. The particular permission is assigned to the role for some instance when no conflicts were found by the algorithm.

1. Fuzzy Condition-Validity-Test

2. Fuzzy Condition-Conflict-Test

3. Obligation-Ambiguity-Test

4. Modified Spatial Temporal Fuzzy Multiple-PA-Conflict-Detection

The algorithm Fuzzy Condition-Validity-Test is used to transform a condition to a scope list and verify the validity of a condition at the same time. If a condition is not satisfied, it is meaningless to compare a permission assignment with the condition in another permission assignment.

**Algorithm** Fuzzy Condition-Conflict-Test (μ*cvl*1, μ*cvl*2)
**Initialisation:**
μ*cvlj* is an arraylist indexed by the unique name of each context variable used in μ*cvaj* , and each element of μ*cvlj* to contain the name, type, and the value scope of the corresponding context variable;
*cv* is a local object to store a context variable information temporally.
1: *conflicting* ← **false**
2: **for all context** variable *cv* in μ*cvl*1 **do**
3: **if** μ*cvl*2.*contain* (*cv.name*) **then**
4: **if** (*cv.scope* ∩ μ*cvl*2 [*cv.name*].*scope*) equals to NULL **then**
5: **if** *cv.type* equals to μ*SCV* **then**
6: return **1**
7: **else**
8: *conflicting* ← **true**

9: **end if**
10: **end if**
11: **end if**
12: **end for**
13: **if** *conflicting* **then**
14: return **2**
15: **else**
16: return **3**
17: **end if**

Obligation-Ambiguity-Test algorithm (as defined by Qun Ni et al [2]) is used to find the conflicts that occur in the permission assignments due to ambiguity in the obligations of the permission assignments.

The return value from each of the algorithm helps in deciding the output. The meaning of each of the return value is provided in Table 1

TABLE 1 RETURN VALUES

| Value | Meaning |
|---|---|
| -1 | Invalid condition that is not satisfiable |
| 0 | No conflict between two conditions |
| 1 | No conflict between two conditions because some shared SCV has an empty scope. It means that two corresponding permissions are aiming at different partitions of data. Here no need of checking ambiguous obligations further. |
| 2 | A conflict between two conditions because some shared context variable has an empty scope |
| 3 | No conflict between two conditions because no shared context variables have an empty scope. However, in this situation we need further check obligation ambiguity before making the final judgement. |
| 4 | No conflict because of the obligations |
| 5 | Conflict caused by ambiguous obligations. |

The Modified Spatial Temporal Multiple-PA-Conflict Detection algorithm takes as input the requested permission assignment and divides it to the atomic level.

**Algorithm** Modified Spatial Temporal Multiple-PA-Conflict-Detection(µ*PA*, µ*pal)*
**Initialisation:**

µ*PA* is the permission assignment that is requested

µ*pal* is the arraylist of all the permission assignments already made

Each of the individual components can be separately accessed as

*role*, *data*, *action*, *purpose*, *condition/fuzzy condition* and *obligation*.

1:result<−FuzzyCondition-Validity-Test (µPA.cndition, µcv1, µcv2... µcvn)
2: **if** result = -1 then
3: **exit** // invalid condition
4: **end if**
5: **for all** µ*pa* such that µ*pa* <−*L*µ*pa* **do**
6: **for** i = 1 to n **do**
7:result<−FuzzyCondition-Conflict-Test (µPA.condition, µpa[i], µcv[i])
8: **if** its result is *equal* to -1 **then**
9: **do begin**
10: **for** j = 1 to n do
11: *Lcp*.add(µpa[j], µcv[j])//conflicting permission
12: **end**
13: **exit**
14: **end if**
15: **end for**
16: **for** i = 1 to n **do**
17: result←Obligation-Ambiguity-Test(µPA.obligation, µpa[i].obligation, µcv[i].obligation)
18: **if** its result is *equal* to -1 **then**
19: **do begin**
20: **for** j = 1 to n **do**
21: *Lcp*.add (µpa[j], µcv[j])
22: **end exit**
23: **end if**
24: **end for**
25: **if** µPA.purpose to≠ µPA.purpose.intended **then**
26: *Lcp*.add(µpa[i], result)
27: **end if**
28: **if** result equals to 1 **then**
29: assg.add(µPA, µcv)
30: **end if**
31: **end for**

If at any of the stage a conflict occurs they are noted and a detailed report is given indicating where the conflict occurs and also with which permission

assignment the conflict occurs. By providing such detailed reports the user and the administrator can make use of it to avoid the conflict and revise any of the existing permission assignment.

### *D. Permission Assignment:*

The permissions are properly assigned to roles by the administrator. The administrator initially does executing the conflict detection module. Based on the results it decides to grant/deny the permission.

### *E.. System Administrator:*

The system administrator maintains the database and controls the entire process of online healthcare system. It stores the information's of the newly arriving patients to the system and properly allocate the permissions to the perfect roles.

## IV. Result and Analysis

We have found the advantage of spatial temporal constraints and fuzzy logic in privacy aware role based access control model that provides the efficient technique to set the range to obtaining the permissions. Thus the users are limited to access the permissions assigned to the roles. The Fuzzy context variable plays the vital role in conflict detection. A simple and flexible conflict detection criteria has been achieved within a short period of
time which is shown in form of a graph. Fig. 4 explains the conflict detection accuracy analysis.

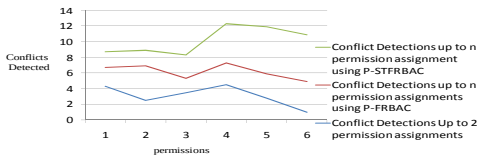


Fig. 4.1 Conflict detection accuracy analysis

## V. Conclusion

In this paper, we have proposed Privacy aware spatial temporal fuzzy based RBAC Model and implemented a modified conflict detection algorithm using fuzzy spliting context variable, spatial and temporal constraints. The proposed algorithm detects the conflicts when there is a conflict between three or more permission assignments in the proposed model. The proposed model also provides effective access control to the users involved in the system based on the location and time constraints.

## References

[1] Carless Martinez-Garcia, Guillermo Navarro-Arribas and Joan Borrel (2011). "Fuzzy Role-Based Access Control".Elsevier Information Processing letters 111, 483-487.

[2] M Caroline Joan, N Jaisankar, A Kannan. (2011) "A Modified Conflict Detection Algorithm for Multiple Permission Assignments in Privacy Aware – RBAC Model"

[3] Qun Ni, Elisa Bertino, Carolyn Brodie, Clare-Marie Karat, John Karat, Jorge Lobo and Alberto Trombetta. (2010) "Privacy aware Role Based Access Control", ACM Transactions on Information and System Security.

[4] Lorenzo D. Martino, Qun Ni, Dan Lin and Elisa Bertino. (2008) "Multi-domain and Privacy-aware Role Based Access Control in eHealth", Conference Proceedings of Second International Conference on Pervasive Computing Technologies for Healthcare.

[5] Hassan Takabi, Morteza Amini and Rasool Jalili (2007) "Enhancing Role Based Access Control Model through Fuzzy Relations" Third International Symposium on Information Assurance and Security, 0-7695-2876.

[6] U.H.G.R.D Nawarathna and S.R.Kodithuwakku (2005). "A Fuzzy Role Based Access Control Model for Database Security". Proceedings of International conference on Information and Automation.

[7] Axel Kern and Claudia Walhorn. (2005) "Rule support for Role Based Access Control"

[8] Frode Hansen and Vladimir Oleshchuk. (2003) "A Spatial Role-Based Access Control Model for Mobile Systems", Conference of Information Security, P.P 129-141.

[9] Elisa Bertino, Piero Andrea Bonatti, Elena Ferrari. (2001) "TRBAC: A Temporal Role-Based Access Control Model", ACM Transactions on Information and System Security, Vol. 4, No.3.

[10] Ravi S.Sandhu, Edward J. Coynek, Hal L.Feinsteink and Charles E.Youmank.(1996) "Role-Based Access Control Models".IEEE Computer, Volume 29, Number 2, pages 38-47.

[11] L. A. Zadeh (1965). "Fuzzy Sets". Information and Control 8, 338-353.