Analysis of TRBAC with Dynamic Temporal Role Hierarchies

Emre Uzun, Vijayalakshmi Atluri, Jaideep Vaidya, Shamik Sural

Analysis of TRBAC with Dynamic Temporal Role Hierarchies

Emre Uzun[1], Vijayalakshmi Atluri[2], Jaideep Vaidya[1] and Shamik Sural[3]

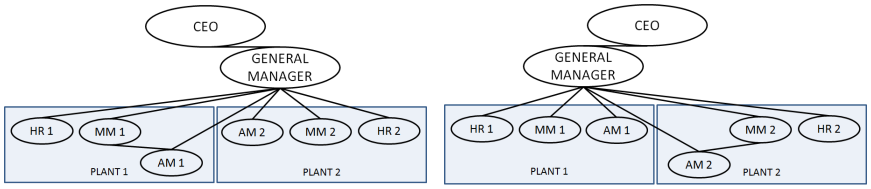
[1] MSIS Department, Rutgers Business School, USA
{emreu,jsvaidya}@cimic.rutgers.edu
[2] National Science Foundation and MSIS Department, Rutgers Business School, USA
atluri@cimic.rutgers.edu
[3] School of Information Technology, IIT Kharagpur, India
shamik@cse.iitkgp.ernet.in

Abstract. The temporal role based access control (TRBAC) models support the notion of temporal roles, user-to-role and permission-to-role assignment, as well as allow role enabling. In this paper, we argue that role hierarchies can be temporal in nature with a dynamism that allows it to have a different structure in different time intervals; and safety analysis of such extensions is crucial. Towards this end, we propose the temporal role based access control model extended with dynamic temporal role hierarchies, denoted as TRBAC_{RH}, and offer an approach to perform its safety analysis. We also present an administrative model to govern changes to the proposed role hierarchy.

1 Introduction

The temporal extension of the role based access control (TRBAC) model assumes one or more of the following features: temporal User to Role Assignments, temporal Permission to Role Assignments, role enabling, and role hierarchies [2, 7]. In this paper, we introduce dynamic temporal role hierarchies for TRBAC. Role Hierarchies (RH), or sometimes called Role to Role Assignments (RRA), are one of the three basic relations that are defined in RBAC along with URA and PRA [9]. Whether the basis for RH in an enterprise is either functional or administrative, it simply allows higher level (senior) roles inherit the permissions assigned to the lower level (junior) roles.

In this paper, we argue that the role hierarchies can be temporal in nature, i.e., they may change with time. Although role hierarchies in prior temporal extensions of RBAC have been specified, they do not allow temporal constraints to be specified on RH that not only *restrict the time* during which the hierarchy is valid, but also *change its structure* by shifting the position of the roles in the hierarchy. Essentially this means that a senior level role cannot always inherit the permissions of a junior level role. Also, a role may change its level in the hierarchy, for example, a junior level role may be elevated to a higher level role during certain time periods. To capture this dynamic structure, we enhance the traditional definition of TRBAC with *Dynamic Temporal Role Hierarchy* (DTRH), and we denote the resulting model, TRBAC_{RH}, a temporal role based access control model with dynamic temporal role hierarchies. Although enterprises usually specify a static hierarchy, DTRH comes into play in some temporary or periodical



(a) Role Hierarchy when General Manager is at Plant 2

(b) Role Hierarchy when General Manager is at Plant 1

Fig. 1. Role Hierarchies on Different Days of the Week

exceptional situations that are required for operational purposes. In the following, we provide such a motivating example.

Consider a manufacturing company with two different production plants, one having the headquarters of the company. The company has a CEO and a General Manager (GM) who works at both the plants; an Accounting Manager (AM), a Manufacturing Manager (MM), and a Human Resources Manager (HR) for each plant. Although CEO works at the headquarters, GM works in both of the plants in different days of the week. As in Figures 1(a) and 1(b), when he is present at a plant, he manages the operations and audits the actions of the AM of that plant. However, when he is at the other plant, MM has the responsibility to audit the operations of AM without completely assuming the GM role, which is considered to have many additional permissions. Since the hierarchical relationships among the roles change, this situation can be specified by DTRH, by simply having a policy which makes MM move to the second level, on top of AM only on the days when GM is away. Nevertheless, it is still possible to represent the scenario in this example using a static role hierarchy. However, lack of temporal role hierarchies will force the system administrators to create a dummy role, like "Manager and Auditor" (MA), that does not essentially represent a regular job function. Also, this role should have the required permission and hierarchy assignments that MM needs. Moreover, MM should be assigned to two separate roles (MM and MA) which are enabled and disabled in regular time intervals. Clearly, creation of such redundant dummy roles increases the administrative burden [4].

Role delegation, is another way of handling such scenarios [3, 12, 1, 11, 5]. Users are delegated to the necessary roles of the users that are away. Although this process seems more practical than dealing with dummy roles, some complications are possible. The *delegatees* might not be allowed to assume all of the permissions of the role that they are delegated. At this point, we have to note that [12] and [5] provides a scheme for partial delegation by either temporary dummy roles or blocking some permissions in the delegated role. Even though our example scenario can be modeled using role delegation without imposing significant overhead, employing temporal role hierarchies has still an advantage as it lends itself for performing *safety analysis* since none of the role delegation studies propose it.

The main contribution of this paper is to perform safety analysis of TRBAC_{RH}. Whether handling the temporal role hierarchies is done using the specification of DTRH, using dummy roles or delegation, none of the prior work on safety analysis considers

RBAC models with temporal constraints on role hierarchies. The safety analysis of TRBAC_{RH} leads us to expand the set of possible safety questions. As discussed above, having DTRH can reduce redundancy and facilitate the administration in various dynamic work environments. Since we have a dynamic hierarchy, which is controlled by an administrative model (Section 3.2), the implicit role assignments require much more attention than before. There is no problem of this sort in the case of static role hierarchies, however a simple manipulation in the hierarchy could create a security breach, and should be detected in advance to prevent any such occurrence. Therefore, we need to examine new security questions in the analysis of systems with dynamic temporal role hierarchies. A possible safety question can be: "Will a user u ever get *implicitly* assigned to role r in the future?" A liveness question can be: "Will a user u ever lose any role that he is *implicitly* assigned in the future?" Finally, a mutual exclusion question can be: "Will users u_1 and u_2 ever get *implicitly* assigned to role r at the same time slot in the future?"

We define the TRBAC_{RH} by extending the definitions of TRBAC with the dynamic temporal role hierarchies, as well as its administrative model. We also propose an approach to perform safety analysis on this model to answer potential safety questions discussed above. For our analysis, we adopt the TRBAC safety analysis approach recently proposed by Uzun et al. [10]. Specifically, we decompose the TRBAC_{RH} analysis problem into multiple RBAC analysis problems and simply employ existing RBAC analysis techniques to solve the TRBAC_{RH} analysis.

2 Preliminaries

Temporal Role Based Access Control Model: Temporal RBAC was first proposed by Bertino et al. [2] to be an RBAC model with the capability of role enabling and disabling via periodical and duration constraints. Joshi et al. [7] extended this model to have temporal capabilities on user to role and role to permission assignments along with some other components like constraints, role triggers and role hierarchies. In both of these models, the time notion is embedded using *Calendar* expression which is composed of *periodicity* and *duration* expressions. Uzun et al. [10] provide a simplified version of the temporal models of [2] and [7] in order to provide strategies to perform safety analysis on TRBAC. The main difference between this model and the models by [2] and [7] is the simplified calendar expression, which only has periodicity constraints. Since we base our temporal role hierarchies on the TRBAC model by [10], we now give some of its components and notation.

Let U, R, $PRMS$ be finite sets of users, roles and permissions, respectively, of a traditional RBAC system. Although the PA relation, $PA \subseteq PRMS \times R$ is defined the same way as in RBAC [9], UA relation is defined in a different way, considering the temporal nature of the model. The unit time is represented by discrete *time slots*. Let T_{MAX} be a positive integer. A *time slot* of $Times$ is a pair $(a, a+1)$, where a is an integer, and $0 \leq a < a+1 \leq T_{MAX}$. We use the term *time interval*, for a consecutive series of time slots. A *schedule* s over T_{MAX} is a set of time slots. The model has the periodicity property (just like the preceding TRBAC models) which is provided by having schedules that repeat themselves in every T_{MAX} time slots. This

temporal notion is embedded into two different components of the model: $TUA \subseteq (U \times R \times S)$ is the *temporal user to role assignment* relation and $RS \subseteq (R \times S)$ is the *role-status* relation which controls the role enabling and disabling. A tuple $(u, r, s) \in TUA$ represents that user u is a member of the role r only during the time intervals of schedule s. A tuple $(r, s) \in RS$ imposes that role r is *enabled* only during the time intervals of s and therefore it can only be assumed at these times. Thus, a user u can assume role r at time $t \in [0, T_{MAX}]$ provided that $(u, r, s_1) \in TUA$, $(r, s_2) \in RS$, and $t \in (s_1 \cap s_2)$, for some schedules s_1 and s_2. The administrative model for TRBAC is used to change these two temporal components. More specifically, the administrative rules `t_can_assign`, `t_can_revoke`, `can_enable` and `can_disable` is used to assign / revoke roles to users, and enable / disable roles, respectively. Applying these rules change the assignments along with their schedules.

Static and Temporal Role Hierarchies: A Role Hierarchy relationship $(r_1 \geq r_2)$ between roles r_1 and r_2 means that r_1 is superior to r_2, so that any user who has r_1 assigned, can inherit the permissions assigned to r_2. In traditional RBAC, this assignment is, naturally, static [9]. However, presence of a temporal dimension brings some additional flexibility on how these hierarchies work. Previously proposed models for temporal role hierarchies [6, 7] focus on the permission and activation inheritance through the role hierarchies in the presence of role enabling and disabling. Particularly, the role hierarchy is still static, but the temporal constraints on the role enabling determines whether the role hierarchy will provide inheritance for a role at a given time. Three types of hierarchy relations for temporal domain are proposed: *Inheritance Only Hierarchy* $(\geq)$, *Activation Only Hierarchy* $(\succeq)$ and *General Inheritance Hierarchy* $(\gg)$. Lastly, a *Hybrid Hierarchy* exists when the pairwise relations among different roles are of different types. Interested readers may consult [6, 7] for details.

3 Dynamic Temporal Role Hierarchies in TRBAC

The flexibility to have a different hierarchy structure at different time intervals makes Dynamic Temporal Role Hierarchy different than the Temporal Role Hierarchy in [7]. In order to represent this additional capability, we provide a new Role to Role Relation called *dynamic temporal role hierarchy policy*, and an administrative model to make modifications on it, like the RRA97 relation of ARBAC97 [8].

3.1 Dynamic Temporal Role Hierarchy Policies

A TRBAC policy with the presence of dynamic temporal role hierarchies, denoted as TRBAC_{RH}, and is defined as follows: Let S be the set of all possible schedules over T_{MAX}. A TRBAC_{RH} policy over T_{MAX} is a tuple $M = \langle U, R, PRMS, TUA, PA, RS, DTRH \rangle$ where $DTRH \subseteq (R \times R \times S \times \{weak, strong\})$ is the *temporal role hierarchy relation*. In our model, $DTRH$ is represented as a collection of dynamic temporal role hierarchy policies, which are tuples consisted of a pair of roles associated with a schedule that denotes the time slots that the policy is valid. In our model, we have dynamic temporal role hierarchy for inheritance only relation $DTRH_I$, for activation only relation $DTRH_A$ and for general inheritance relation $DTRH_{IA}$. For notational simplicity, we use $DTRH$, when we refer to any one of them.

Definition 1. *A dynamic temporal role hierarchy policy* $(r_1 \geq_{s,weak} r_2) \in DTRH_I$ *between roles* r_1 *and* r_2 *is an inheritance-only weak temporal relation, that is valid in the time slots specified by a schedule* s*. Under this policy, a user* u *who can activate* r_1 *can inherit permissions of* r_2 *at time* t *if (1)* $(u, r_1, s_1) \in TUA$ *(2)* $(r_1, s_2) \in RS$ *and (3)* $t \in (s_1 \cap s_2 \cap s)$*, provided that there exists schedules* s_1 *and* s_2 *that determine the time slots that* u *is assigned to* r_1 *and* r_1 *is enabled, respectively.*

Definition 2. *A dynamic temporal role hierarchy policy* $(r_1 \succeq_{s,weak} r_2) \in DTRH_A$ *between roles* r_1 *and* r_2 *is an activation-only weak temporal relationship, that is valid in the time slots specified by a schedule* s*. Under this policy, a user* u *can activate* r_2 *at time* t *if (1)* $(u, r_1, s_1) \in TUA$ *(2)* $(r_2, s_2) \in RS$ *and (3)* $t \in (s_1 \cap s_2 \cap s)$*, provided that there exists schedules* s_1 *and* s_2 *that determine the time slots that* u *is assigned to* r_1*, and* r_2 *is enabled, respectively.*

Definition 3. *A dynamic temporal role hierarchy policy* $(r_1 \gg_{s,weak} r_2) \in DTRH_{IA}$ *between roles* r_1 *and* r_2 *is a general weak temporal relationship, that is valid in the time slots specified by a schedule* s*. Under this policy, a user* u *can activate* r_2 *at time* t*, or inherit permissions of* r_2 *if (1)* $(u, r_1, s_1) \in TUA$ *(2)* $(r_2, s_2) \in RS$ *and (3)* $t \in (s_1 \cap s_2 \cap s)$*, provided that there exists schedules* s_1*, and* s_2 *that determine the time slots that* u *is assigned to* r_1 *and* r_2 *is enabled, respectively.*

In the above three definitions, the relations become strong, (i.e: $r_1 \geq_{s,strong} r_2) \in DTRH_I$, $(r_1 \succeq_{s,strong} r_2) \in DTRH_A$ and $(r_1 \gg_{s,strong} r_2) \in DTRH_{IA}$), when (2) is replaced with $(r_1, s_2), (r_2, s_3) \in RS$ and (3) is replaced with $t \in (s_1 \cap s_2 \cap s_3 \cap s)$ where s_3 is the schedule that determine the time slots that r_2 is enabled. Now, let us give an example about how these policies work.

Consider that we have temporal access control system with three roles, r_1, r_2 and r_3 and $T_{MAX} = 3$. Suppose that we have the following $DTRH$ and RS policies defined: (1) $(r_1, (0, 2)) \in RS$ (2) $(r_2, (0, 1)) \in RS$ (3) $(r_3, (1, 3)) \in RS$ (4) $(r_1 \geq_{(0,3),strong} r_2) \in DTRH_I$ (5) $(r_1 \geq_{(0,3),weak} r_3) \in DTRH_I$. According to these policies, a user who has r_1 assigned can inherit permissions of r_2 only in the time interval $(0, 1)$, because r_2 is not enabled in $(1, 3)$ and the role hierarchy relation is strong. However, u can inherit permissions of r_3 in $(0, 2)$, even if r_3 is not enabled in $(0, 1)$, since the relation is weak.

A *hybrid* relation in a dynamic temporal role hierarchy, $DTRH_H$, may contain all of the tuples defined in the above definitions, and each relation among different roles is determined using the type of that specific relation.

Dynamic temporal role hierarchy policies $(r_1 \geq_{s,weak} r_2) \in DTRH$ satisfy the following properties for a given schedule s: (1) Reflexive: $(r_1 \geq_{s,weak} r_1) \in DTRH$, (2) Transitive: If $(r_1 \geq_{s,weak} r_2), (r_2 \geq_{s,weak} r_3) \in DTRH$, then $(r_1 \geq_{s,weak} r_3) \in DTRH$. (3) Asymmetric: If $(r_1 \geq_{s,weak} r_2) \in DTRH$ then $(r_2 \geq_{s,weak} r_1) \notin DTRH$. These properties apply for both strong and the other types of relations $(\succeq, \gg)$ as well.

3.2 Administrative Model for TRBAC with Dynamic Temporal Role Hierarchies

Administrative models are required for RBAC systems in order to govern the modifications on the access control policies [8]. Without these models, the access control

policies are considered static as $DTRH$, which is a static policy unless there is an administrative model to allow for modifications. Uzun et al.[10] present an administrative model for TRBAC. In this section, we propose an extension to that model, which makes it cover TRBAC$_{\mathrm{RH}}$. This extension is composed of a rule called `t_can_modify`, similar in semantics to the `can_modify` in [8], but with additional capabilities for temporal dimension. This rule updates the valid time slots of the dynamic temporal role hierarchy policies. Also, in contrast to precondition structures that have been proposed in the literature for other administrative rules (like `can_assign`), it has two sets of preconditions, one for senior and one for junior role in order to protect the integrity of the hierarchy. The rule is composed of eight parameters that should be satisfied to execute the rule. Let t be the time slot that the rule is required to be executed. (1) $admin$ denotes the administrative role that a user must belong in order to execute the rule. (2) s_{rule} is a schedule that denotes the time slots in which the rule is executable. In order to satisfy, $t \subseteq s_{rule}$. (3) $s_{hierarchy}$ is a schedule that denotes the time slots of the hierarchy policy that the rule is authorized to modify. (4) $type \in \{strong, weak\}$ denotes the type of the hierarchy relation. (5) r_{sr} is the senior role of the hierarchy policy. (6) r_{jr} is the junior role of the hierarchy policy. (7) $SR(Pos, Neg)$ denotes the positive and negative preconditions of the senior role r_{sr}. The preconditions are satisfied in the following way: Let $\hat{s}$ denote the time slots that are intended to be modified by the rule ($\hat{s} \subseteq s_{hierarchy}$). For each $r \in Pos$, there must be a role hierarchy policy $(r \geq_{\hat{s},type} r_{sr}) \in DTRH$ and for each $r \in Neg$, there must not be a hierarchy policy $(r \geq_{\hat{s},type} r_{sr}) \in DTRH$. (8) $JR(Pos, Neg)$ denotes the positive and negative preconditions of the junior role r_{jr}. The preconditions are satisfied in the following way. Let $\hat{s}$ denote the time slots that are intended to be modified by the rule ($\hat{s} \subseteq s_{hierarchy}$). For each $r \in Pos$, there must be a role hierarchy policy $(r_{jr} \geq_{\hat{s},type} r) \in DTRH$ and for each $r \in Neg$, there must not be a hierarchy policy $(r_{jr} \geq_{\hat{s},type} r) \in DTRH$. Under these parameters, a tuple $(admin, s_{rule}, SR(Pos, Neg), JR(Pos, Neg), s_{hierarchy}, r_{sr}, r_{jr}, type) \in$ `t_can_modify` allows to update the role hierarchy relation $r_{sr} \geq_{s,type} r_{jr}$ as follows: Let $\hat{s}$ be a schedule over T_{MAX} with $\hat{s} \subseteq s_{hierarchy}$. Then, if this rule can be executed at time t, and the preconditions are satisfied w.r.t. schedule $\hat{s}$, then the tuple $r_{sr} \geq_{s,type} r_{jr}$ is updated to $r_{sr} \geq_{s \cup \hat{s},type} r_{jr}$ or $r_{sr} \geq_{s \setminus \hat{s},type} r_{jr}$, depending on the intended modification. This definition is for inheritance only hierarchies, but it also applies to activation only and general inheritance hierarchies, by replacing $\geq$ with $\succeq$ and $\gg$.

4 Toward Safety Analysis of TRBAC Systems with Dynamic Temporal Role Hierarchies

An important aspect of any access control model is its safety analysis. This is necessary to answer security questions such as those given in Section 1. In this section, we examine how safety analysis of the TRBAC model with DTRH can be carried out. The basic idea is to use the decomposition approach proposed in [10], which reduces the TRBAC safety problem into multiple RBAC safety sub-problems and handles each sub-problem separately using an RBAC safety analyzer that has been proposed in the literature. Here, we need to make two assumptions: (1) The administrative model for the dynamic temporal role hierarchy given in Section 3.2 cannot completely be decom-

posed into a traditional RBAC. The underlying reason is the precondition structure of dynamic temporal role hierarchies, that does not exist in ARBAC97 role hierarchy component RRA97. Decomposing TRBAC into multiple RBAC safety sub-problems relax the schedule components, but the precondition requirements remain in effect. Since, there is no known RBAC safety analyzer that can handle preconditions in role hierarchies, we assume that the administrative model for dynamic temporal role hierarchy contains rules with no precondition requirement for safety analysis purposes. (2) The safety questions and the structure of the analysis in [10] are based on checking the presence of a particular role (or roles) being assigned to a particular user. There is no permission level control available in the model. Hence, we restrict our safety analysis on the Activation-Only hierarchies. So, we assume $DTRH = DTRH_A$. Moreover, we assume all relationships are strong.

The decomposition that we utilize is the Role Schedule Approach of [10]. In this approach the sub-problems are constructed using the role schedules of the administrative rules. In TRBAC, the administrative rules for role assignment and role enabling have two separate schedules: Rule Schedule and Role Schedule. Rule schedule is similar to the s_{rule} of `t_can_modify` and determines the periods in which the rule is valid. Similarly, the role schedule is similar to the $s_{hierarchy}$ of the `t_can_modify` and determines the time slots that the rule is authorized to modify TUA and RS policies. The key observation that makes this decomposition possible is the independency among different time slots, and the periodic behavior of the model. Particularly, if we are interested in the safety analysis of a time slot t, then we only need to consider the administrative rules, that are authorized to modify time slot t of TUA and RS relations. Furthermore, since the system is periodic, for any long run analysis, we can safely assume that the validity constraints of the rules (s_{rule}) will be enforced implicitly even if they are ignored. For detailed discussion, readers may refer to [10]. In TRBAC_{RH}, we perform similar operations to generate sub-problems. Our model has the same property of having independency among the time slots. First, we define the dynamic temporal role hierarchy policies for a single time slot t, denoted as $DTRH_t \subseteq DTRH$ to be a collection of role hierarchy policies in the system that satisfies:

$$DTRH_t = \{(r_i \succeq_{s_1,strong} r_j) | t \in s_1\} \forall r_i, r_j, \in R$$

where s_1 and s_2 are the schedules of role hierarchy rules. The dynamic role hierarchy policies in the time slot t of RH_t, then reduces to $(r_i \succeq_{strong} r_j)$. These policies become non-temporal role hierarchy policies for the time slot t. Similarly, administrative rules for a specific time slot t is defined as:

$$(admin, s_{rule}, SR(\emptyset, \emptyset), JR(\emptyset, \emptyset), s_{hierarchy}, r_{sr}, r_{jr}, type) \in \texttt{t\_can\_modify}_t, t \in s_{hierarchy}$$

Hence, the set $\texttt{t\_can\_modify}_t$ contains the administrative rules that are authorized to modify the t^{th} time slot of the dynamic temporal role hierarchy policies. In other words, if one is interested in the safety analysis for time slot t, then there is not any other administrative rule authorized to modify time slot t, but the rules in $\texttt{t\_can\_modify}_t$. The administrative rules for t are reduced to $(admin, r_{sr}, r_{jr}, type)$ which belong to the `can_modify` $\subseteq admin \times 2^R$ of the RRA97 of [8].

The above defined two sets, RH_t and $\texttt{t\_can\_modify}_t$ provide safety analysis using the Role Schedule Approach in [10]. When decomposed, there are k RBAC safety sub-problems, where k is the number of time slots. Any RBAC safety analyzer that is capable of handling role hierarchies and the `can_modify` relation can be used to analyze these sub-problems. Repeating this operation for each of these k sub-problems will yield the safety analysis of the TRBAC_{RH} system. The computational complexity of this process depends linearly on the computational complexity of the RBAC safety analyzer and k.

5 Conclusion and Future Work

In this paper, we introduced the concept of dynamic temporal role hierarchy, which, can be viewed as a different role hierarchy at different times. We develop an administrative model for RBAC with dynamic temporal role hierarchies along with a road map for its safety analysis. Currently we are implementing safety analysis using the RBAC analysis tools available. Implementing administrative model of the dynamic temporal role hierarchies in the safety analysis will require new tools to fully capture the capabilities of the model. This will be our future work.

Acknowledgements: This work is partially supported by the National Science Foundation under grant numbers CNS-0746943 and CNS-1018414.

References

1. E. Barka, R. Sandhu, et al. A role-based delegation model and some extensions. In *NISSC*, volume 4, pages 49–58, 2000.
2. E. Bertino, P.A. Bonatti, and E. Ferrari. TRBAC: A temporal role based access control model. *ACM Transactions on Information and System Security*, 4(3):191–233, 2001.
3. J. Crampton and H. Khambhammettu. Delegation in role-based access control. *Computer Security–ESORICS 2006*, pages 174–191, 2006.
4. Q. Guo, J. Vaidya, and V. Atluri. The role hierarchy mining problem: Discovery of optimal role hierarchies. In *ACSAC 2008*, pages 237–246. IEEE, 2008.
5. J.B.D. Joshi and E. Bertino. Fine-grained role-based delegation in presence of the hybrid role hierarchy. In *SACMAT*, pages 81–90, 2006.
6. J.B.D. Joshi, E. Bertino, and A. Ghafoor. Hybrid role hierarchy for generalized temporal role based access control model. In *COMPSAC 2002*, pages 951–956. IEEE, 2002.
7. J.B.D. Joshi, E. Bertino, U. Latif, and A. Ghafoor. A generalized temporal role based access control model. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):4–23, 2005.
8. R. Sandhu, V. Bhamidipati, E. Coyne, S. Ganta, and C. Youman. The ARBAC97 model for role-based administration of roles: preliminary description and outline. In *ACM Workshop on Role-Based Access Control*, pages 41–50, 1997.
9. R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.
10. E. Uzun, V. Atluri, S. Sural, J. Vaidya, G. Parlato, A. Ferrara, and M. Parthasarathy. Analyzing temporal role based access control models. In *SACMAT*. ACM, 2012.
11. L. Zhang, G.J. Ahn, and B.T. Chu. A rule-based framework for role-based delegation and revocation. *TISSEC*, 6(3):404–441, 2003.
12. X. Zhang, S. Oh, and R. Sandhu. PBDM: a flexible delegation model in rbac. In *SACMAT*, pages 149–157. ACM, 2003.