Risk-Aware Role-Based Access Control

Liang Chen and Jason Crampton

Information Security Group and Department of Mathematics
Royal Holloway, University of London
{liang.chen.2005,jason.crampton}@rhul.ac.uk

Abstract. The increasing need to share information in dynamic environments has created a requirement for risk-aware access control systems. The standard RBAC model is designed to operate in a relatively stable, closed environment and does not include any support for risk. In this paper, we explore a number of ways in which the RBAC model can be extended to incorporate notions of risk. In particular, we develop three simple risk-aware RBAC models that differ in the way in which risk is represented and accounted for in making access control decisions. We also propose a risk-aware RBAC model that combines all the features of three simple models and consider some issues related to its implementation. Compared with existing work, our models have clear authorization semantics and support richer types of access control decisions.

1 Introduction

Access control mechanisms are typically policy-based, meaning that attempts to access resources are allowed or denied based on whether the access is authorized by some policy. Traditionally, the job of an access control system is to decide whether an access is authorized and to allow only those access attempts that are authorized.

Risk-aware access control is a novel access control paradigm that was proposed to meet the increasing need to share information in "agile" and ephemeral organizations such as coalitions and collaborations [7,14,16,18,24]. The core goal of developing risk-aware access control is to provide a mechanism that can manage the trade-off between the risk of allowing an unauthorized access with the cost of denying access when the inability to access resources may have profound consequences. When a user makes a request to access some resources, a risk-aware access control mechanism will evaluate the request by estimating the expected costs and benefits of granting access: the request might be denied if the risk is above some system-defined threshold; alternatively, the request might be denied if the cost exceeds the expected benefit.

This approach to deciding access requests is completely different from earlier access control models in which access control decisions are made on the basis of predefined policies that explicitly distinguish between allowed and denied access. In other words, risk-aware access control systems are designed to be more permissive than traditional access control mechanisms, in the sense that some

C. Meadows and C. Fernández-Gago (Eds.): STM 2011, LNCS 7170, pp. 140–156, 2012.

risky or exceptional accesses are allowed, provided the risk of allowing such access can be accounted for and is not unacceptably high. Therefore, an important step for modeling risk-aware access control is to identify appropriate ways of estimating and managing risk. Most existing work in the literature attempts to achieve this in the context of multi-level security [7,8,18].

Role-based access control (RBAC) has been the subject of considerable research in the last decade [3,12,23], resulting in the release of the ANSI RBAC standard [1]. A number of commercial products, such as Windows Authorization Manager and Oracle 9, implement some form of RBAC. The basic idea of RBAC is that users and permissions are associated with roles, and that there are significantly fewer roles than there are users or permissions. In other words, a role provides a convenient way of associating a group of users with some set of permissions. This feature of role abstraction greatly simplifies the management of access control policies. Some other features that make RBAC attractive include the support for role hierarchies and the specification of separation of duty constraints [1].

To make use of RBAC in dynamic environments, we believe that there is a pressing need to develop appropriate risk-aware RBAC models, and it is this need we address in this paper. However, existing work in this area has one common limitation: existing models for risk-aware RBAC [2,5,11,19] only support the type of binary decisions, where the accesses with acceptable risk are allowed (and denied otherwise). We believe that risk-aware RBAC models should be able to make access control decisions on the basis of estimates of risk, system-defined risk thresholds, and risk mitigation strategies. In addition, existing models usually assess risk of granting access requests in terms of the trustworthiness of users [5,11]. However, the question of whether we could assess risk in terms of other components of RBAC model, to the best of our knowledge, has not yet been adequately investigated. Such considerations are the focus of this paper. More specifically, the contributions of this paper are as follows.

- We argue that the risk of granting an access request in an RBAC system depends on one or more of the following factors: user trustworthiness, the degree of competence of a user with respect to a particular user-role assignment, and the degree of appropriateness of a permission-role assignment for a given role.
- We propose a novel approach to the management and coordination of risk in an RBAC system. Our approach requires that each permission is associated with a risk mitigation strategy that is a list of risk interval and obligation pairs, where each risk interval is associated with an obligation. This approach of defining risk management at the permission level is much more fine-grained than most existing approaches that typically adopt a global mitigation and risk management strategy [7].
- We develop three simple risk-aware RBAC models, varying in the way of measuring and computing risk. These three models augment the standard RBAC96 model with a risk-aware authorization decision function.

– We introduce a risk-aware RBAC model that combines all the features of three simple models, and propose a strategy for the implementation of the model in practice.

The rest of the paper is organized as follows. In the next section we introduce relevant background material, including a graph-based formalism of RBAC96 and our recent work on spatio-temporal RBAC models; this prior work forms the basis for our risk-aware model. In Sect. 3 we discuss how to determine the risk of granting access requests in an RBAC system. In Sect. 4 we formally define the RBAC_T, RBAC_C, RBAC_A models. In Sect. 5 we introduce the full risk-aware RBAC model. Section 6 compares our work with related work in the literature. Section 7 concludes the paper with some suggestions for future work.

2 Background

There are several role-based access control models in the literature, but the best known is undoubtedly the RBAC96 family of models due to Sandhu *et al* [23]. RBAC96 defines four access control models: RBAC_0, RBAC_1, RBAC_2 and RBAC_3. The material in this paper is developed in the context of RBAC_1, which is the most widely used model and corresponds to the hierarchical model in the ANSI RBAC standard; hereafter we write RBAC96 to mean RBAC_1 only.

The RBAC96 model defines a set of roles R, a role hierarchy $RH \subseteq R \times R$, a user-role assignment relation $UA \subseteq U \times R$ (where U is a set of users), and a permission-role assignment relation $PA \subseteq P \times R$ (where P is a set of permissions[1]). We write $\leqslant$ to denote the transitive, reflexive closure of the RH relation; $(R, \leqslant)$ is a partially ordered set (since the directed graph of the role hierarchy relation is assumed to be acyclic). We represent an RBAC96 state (an instance of the RBAC96 model) as a tuple (UA, PA, RH). The RBAC96 state is used to determine whether an access control request (modeled as a user-permission pair) is authorized.

2.1 RBAC96 State as a Directed Graph

We noted that it is convenient to represent an RBAC96 state as an acyclic directed graph [6]. In particular, it provides a simple way of evaluating access requests in an RBAC96 system. In this paper, we will develop our risk-aware RBAC models based on this graph-based representation.

An RBAC96 state (UA, PA, RH) is represented by an acyclic, directed graph $G = (V, E)$, where $V = U \cup R \cup P$, and $E = UA \cup PA \cup RH$. In other words, each vertex v represents an entity, such as a user u, a role r or a permission p in an

[1] The RBAC96 model treats permissions as "uninterpreted symbols", because the precise natural of permissions is "implementation and system dependent". In the ANSI RBAC standard [1], which is based on the RBAC96 model, permissions are defined by an object and an action. For the sake of consistency with the ANSI RBAC standard, we define permissions as object-action pairs in this paper.

RBAC96 system, and each directed edge $e = (v_i, v_j)$ represents a relationship between two entities v_i and v_j; specifically, $(v_i, v_j) \in E$ if and only if (precisely) one of the following conditions holds

$$(v_i, v_j) \in UA, \quad (v_j, v_i) \in RH, \quad (v_j, v_i) \in PA.$$

An *authorization path* (or *au-path*) between v_1 and v_n is a sequence of vertices $v_1, \ldots, v_n$ such that $(v_i, v_{i+1}) \in E$, $i = 1, \ldots, n-1$. In particular, a user u can ("is authorized to") activate a role r if there is an au-path between u and r; a role r is authorized for permission p if there is an au-path between r and p; and a user u is authorized for permission p if there is an au-path between u and p. In other words, determining whether a user u is authorized to invoke a permission p reduces to finding a path from u to p that includes a role activated by u. A central notion in RBAC96 is that of sessions. For ease of exposition, we do not consider sessions until Section 5.1.

2.2 Spatio-Temporal Constraints and Inheritance in RBAC

Recently, we developed a family of spatio-temporal RBAC models [6]. The syntax of these models uses a simple extension of the RBAC96 model with two spatio-temporal constraint specification functions. The semantics of these models are based on the graph-based formalism of RBAC96 described above, and vary in the extent to which RBAC entities and relations are constrained by spatio-temporal restrictions. The state of a spatio-temporal RBAC model is a labeled directed graph, in which $\lambda : V \to D$ and $\mu : E \to D$, for some spatio-temporal domain D. More specifically, the semantics of the *standard model* ($\text{RBAC}^{=}_{ST}$) are determined by the values of λ for nodes on the au-path; the semantics of the *strong model* (RBAC^{+}_{ST}) are determined by λ and μ for nodes and edges on the au-path; the semantics of the *weak model* (RBAC^{-}_{ST}) are determined by the values of λ for the two end-points of the au-path.

We believe that our examination of spatio-temporal RBAC models provides useful insights into the way of developing risk-aware RBAC models. In particular, based on the lessons learned from our study of the complex interactions between constraints and inheritance, we decide not to associate risk with roles and relationships within the role hierarchy. Indeed, it is not clear to us how risk can meaningfully be associated with roles and the role hierarchy. In the next section, we describe how we choose to define risk in RBAC.

3 Defining Risk in RBAC

In the field of information security, the notion of *risk* is often defined in terms of a combination of the likelihood that a threat will occur, and the severity of the resulting impact of the threat [17]. In the context of RBAC, we take the view that the risk of granting a request to perform some permission can be generally determined by the cost that will be incurred if the permission is authorized and subsequently misused, and the likelihood of the permission being misused.

One of the key steps to deploying a risk-aware RBAC system is to estimate the cost of permissions being misused, where the cost of a permission misuse depends on the value of the object associated with that permission and the action that is taken on the object. In the context of multi-level security models, for example, the cost of a read permission being misused is determined by the value of the information requested to be read, where the value of the information is represented by a sensitivity label to which the information is assigned [7]. There exist approaches to determine the cost of permission misuse [15], although choosing an appropriate approach for estimating the cost of permission misuse is likely to be a delicate task.

On the other hard, determining the likelihood of misuse of permissions is inherently hard, since it requires an ability to predict future actions of the requestors. In RBAC, we believe that there are at least three ways in which permission misuse might arise and the likelihood of misuse might therefore be quantified.

Firstly, there is a natural correspondence between *trustworthiness* and the likelihood of misuse of permissions. In earlier RBAC models (and access control models in general), authorized users are always trusted not to misuse the permissions for which they are authorized. Clearly, however, some authorized users are not *trustworthy*. It is reasonable to define different degrees of trustworthiness of users which directly reflects the likelihood of those users to misuse their granted permissions. Intuitively, a user who has a high likelihood of misusing her permissions can be simply regarded as being less trustworthy.[2] In Sect. 4.2 we consider an RBAC model in which user trustworthiness is an explicit part of the model.

In RBAC, a user is authorized for a permission by virtue of role assignments, where roles typically correspond to various job functions within an organization. Users are assigned to roles based on their qualification or competence, whereas permissions are associated with roles based on work-related or functional considerations. In Sect. 4.3, we propose an approach in which the user's competence to perform a role to which she is assigned is explicitly qualified.

Based on the above observations, we assume that if a role is assigned to a less competent user, then this user has a higher likelihood of misusing permissions associated with the role than some more competent user. Conversely, we can try to quantify the degree to which it is appropriate to assign a permission to a role. In some situations, it may be useful to associate a role with permissions for which the role is not obviously appropriate. In a healthcare system, for example, we might wish to restrict authorization to read medical records to doctors and consultants. However, we may choose to authorize the nurse role to read patients' medical records, not because it is appropriate, but because in an emergency it may be vital that there is some healthcare professional who is able to access medical records. Given a role that is associated with some less appropriate permissions with regard to the role's job duties, we take the view

[2] Trust-management and reputation-management system are tools that might be used to compute the trustworthiness of users [9].

that any user who is authorized for this role has a relatively high likelihood of misusing those permissions. We introduce an RBAC model in Sect. 4.4 which explicitly quantifies the appropriateness of a permission-role assignment.

4 Simple Models for Risk-Aware RBAC

In this section, we develop three simple models for risk-aware RBAC that support richer types of policy decisions beyond the usual "deny" and "grant" ones. In particular, the access control decision function in our models is able to make its access decisions based on the RBAC policies and the risk of granting access requests. As discussed in Sect. 3, the risk of granting a request (u, p) is defined using the cost of p misuse, and the likelihood of u to misuse p. There are three distinct possibilities that can be used to measure the likelihood of misuse of permissions, which are embodied in our different models.

4.1 Risk Mitigation

To devise a risk-evaluation strategy for RBAC, we need to determine a *risk threshold* that the RBAC system is willing to accept when granting access requests, and what kind of risk mitigations should occur if a risky access is allowed. In this paper, we define risk thresholds and risk mitigation strategies on a per-permission basis, which provides far more control than the common alternative of setting risk thresholds that apply to all permissions.

We assume the existence of a risk domain $\mathcal{D} = [0, 1] \stackrel{\text{def}}{=} \{d \in \mathbb{R} : 0 \leqslant d \leqslant 1\}$. We write $[t, t')$ to denote the *risk interval* $\{x \in \mathcal{D} : t \leqslant x < t'\}$. Let B denote a set of obligations, where an obligation $b \in B$ is some action(s) that must be taken by the Policy Enforcement Point (PEP) when enforcing an access control decision (as in XACML [20]). For uniformity of exposition, we write $\perp$ to denote the "null" obligation; the PEP is not required to do anything for the null obligation.

Informally, we associate a permission p with a list of interval-obligation pairs: if the risk associated with access request (u, p) is t then we enforce the obligations corresponding to the interval containing t. More formally, we define a *risk mitigation strategy* to be a list $[(0, \perp), (t_1, b_1), \ldots, (t_{n-1}, b_{n-1}), (t_n, \perp)]$, where $0 < t_1 < t_2 < \cdots < t_n \leqslant 1$ and $b_i \in B$. Each permission p is associated with a risk mitigation strategy. Then,

- the request (u, p) is permitted (unconditionally) if the risk of allowing (u, p) is less than t_1;
- the request (u, p) is permitted but the PEP must enforce obligation b_i if the risk of allowing (u, p) belongs to $[t_i, t_{i+1})$;
- the request (u, p) is denied if the risk of allowing (u, p) is greater than or equal to t_n.

It can be seen that the first element of the risk mitigation strategy is redundant; we include it for clarity of exposition.

Although our approach increases the complexity of risk management at the permission level, it is much more fined-grained than most existing approaches that usually adopt a global mitigation and risk management strategy [7,14,16]. In other words, unlike our approach, the occurrence of errors in the management of the global risk thresholds will have an impact on the correctness of deciding all relevant access requests.

In addition, by associating risk thresholds with permissions we simplify the computation of risk of granting requests. In particular, we can ignore the cost of a permission p being misused when considering the risk of granting p. This is because system administrators have "valued" the cost of p's misuse by defining risk thresholds and risk mitigations for p. In contrast, existing approaches that manage risk globally have to be aware of the cost of permissions being misused when evaluating requests for those permissions. Henceforth, we are only concerned with the likelihood of p's misuse in the computation of the risk of granting p.

4.2 The RBAC$_T$ Model

The trustworthiness- and role-based access control (or RBAC$_T$) model augments the standard RBAC96 model with two functions $\alpha : U \to (0, 1]$ and $\lambda : P \to M$, where $\alpha(u)$ denotes the degree of trustworthiness of u, M denotes the set of risk mitigation strategies and $\lambda(p)$ denotes the risk mitigation strategy associated with p's usage. To compute the risk of granting a request (u, p), we define a risk function $risk_T : U \times P \to [0, 1]$ as

$$risk_T(u,p) = \begin{cases} 1 - \alpha(u) & \text{if there exists an au-path from } u \text{ to } p \\ 1 & \text{otherwise.} \end{cases}$$

In other words, $risk_T(u, p)$ is 1 for request (u, p) if there does not exist an au-path from u to p. By definition, for any permission p, the request (u, p) will be denied if the risk of granting it equal to 1; that is, if there is no au-path from u to p. In contrast, if there exists an au-path from u to p, the risk of granting (u, p) is determined by u's trustworthiness. For example, given two requests (u_1, p) and (u_2, p), $risk_T(u_1, p) < risk_T(u_2, p) < 1$ means that allowing u_2 to perform p is more risky than allowing u_1 to access (because u_1 is more trustworthy than u_2). Note that $risk_T(u, p)$ is determined, in part, by the existence of an au-path from u to p. In other words, our approach to risk computation in RBAC$_T$ incorporates the standard RBAC method of checking whether a request is authorized. This will be a common feature of our risk-aware models.

We now define an authorization decision function $Auth_T$ which, given an RBAC$_T$ state (V, E, α, λ), an access request (u, p) and a risk mitigation strategy $\lambda(p) = [(0, \perp), (t_1, b_1), \ldots, (t_{n-1}, b_{n-1}), (t_n, \perp)]$ for p, returns an authorization decision and an obligation. Specifically,

$$Auth_T((V, E, \alpha, \lambda), (u, p), \lambda(p)) = \begin{cases} (\texttt{allow}, \perp) & \text{if } risk_T(u, p) < t_1, \\ (\texttt{allow}, b_i) & \text{if } risk_T(u, p) \in [t_i, t_{i+1}), \\ (\texttt{deny}, \perp) & \text{if } risk_T(u, p) \geqslant t_n. \end{cases}$$

In other words, a request by u to perform p is allowed if there exists an au-path from u to p in the RBAC$_T$ graph and the risk of granting (u, p) is less than a specified risk threshold t_n of p (and denied otherwise). In addition, some system obligations b_i will be forced to execute with the allow access if the risk is perceived as being relatively high (within some interval $[t_i, t_{i+1})$, where $1 \leqslant i < n$).

We believe that the concept of trustworthiness and the risk-assessment methodology we developed in RBAC$_T$ can be naturally integrated into other access control models, enabling them to become risk-aware. In the next section, we introduce risk-aware RBAC models with consideration of competence and appropriateness in the user-role assignments and the permission-role assignments respectively.

4.3 The RBAC$_C$ Model

The competence- and role-based access control (or RBAC$_C$) model augments the standard RBAC96 model with two functions $\beta : U \times R \to (0, 1]$ and $\lambda : P \to M$, where $\beta(u, r)$ denotes u's degree of competence to perform role r, and $\lambda(p)$ denotes the risk mitigation strategy associated with p's usage. Note that, for all $(u, r) \in UA$, we require $\beta(u, r) > 0$; this is because it is not meaningful to assign u to r if u has no competence to perform the role r. Unlike RBAC$_T$, RBAC$_C$ defines the concept of competence on the user-role assignments, which leads to a different way of computing the risk of granting requests.

Given an RBAC$_C$ state $G = (V, E, \beta, \lambda)$, we write $(v, *)$ for the set of entities that are connected *from* v by edges; that is, $(v, *) = \{v' \in V : (v, v') \in E\}$. We also write $(*, v)$ for the set of entities that connected *to* v by edges; that is, $(*, v) = \{v' \in V : (v', v) \in E\}$. For brevity, we write $v*$ for $(v, *)$ and $*v$ for $(*, v)$. Given $v \in V$, we write $\downarrow v$ to denote the set of entities for which v is RBAC96-authorized; that is $\downarrow v = \{v' \in V :$ there exists an au-path from v to $v'\}$. Analogously, we define $\uparrow v = \{v' \in V :$ there exists an au-path from v' to $v\}$.

Given a request (u, p), there may be multiple paths between u and p in the RBAC$_C$ graph. Obviously, we are interested in finding the set of roles for which u is explicitly authorized and that lie on an au-path from u to p, that is $u* \cap \uparrow p$. To compute the risk of granting (u, p), we need to consider the degree of competence that u has to perform each role in $u* \cap \uparrow p$. Given a request (u, p), there might exist one or more au-paths from u to p, the risk of granting (u, p) is determined by finding an au-path $u, r, \ldots, p$ such that $\beta(u, r)$ is maximum. In other words, u is competent to perform all roles in $u* \cap \uparrow p$ to some extent, and there is a role for which she is most competent, therefore this role is the one that is considered when evaluating the access request. Formally, we define a risk function $risk_C : U \times P \to [0, 1]$ as

(a) A simple RBAC$_C$ state (b) A simple RBAC$_A$ state

Fig. 1. A graphical representation of RBAC$_C$ and RBAC$_A$ states

$$risk_C(u,p) = \begin{cases} 1 & \text{if } u* \cap \uparrow p = \emptyset, \\ 1 - \max\{\beta(u,r) : r \in u* \cap \uparrow p\} & \text{otherwise.} \end{cases}$$

Consider the directed graph of an RBAC$_C$ configuration shown in Fig. 1(a), where $\beta(u_1, r_1) = \beta(u_2, r_3) = \frac{1}{2}$, and $\beta(u_1, r_2) = \beta(u_2, r_2) = \frac{1}{3}$. Then u_1 is able to perform p_1 through the role r_1 for which u_1 is most competent. Hence, $risk_C(u_1, p_1) = 1 - \frac{1}{2} = \frac{1}{2}$. However, $risk_C(u_1, p_3) = 1$ as $u* \cap \uparrow p_3 = \{r_1, r_2\} \cap \{r_3\} = \emptyset$ which means there is no au-path from u_1 to p_3.

Given an RBAC$_C$ state (V, E, β, λ), an access request (u, p) and a risk mitigation strategy $\lambda(p)$ for p, we can define an authorization decision function $Auth_C$ (as we did for $Auth_T$):

$$Auth_C((V,E,\beta,\lambda),(u,p),\lambda(p)) = \begin{cases} (\texttt{allow}, \perp) & \text{if } risk_C(u,p) < t_1, \\ (\texttt{allow}, b_i) & \text{if } risk_C(u,p) \in [t_i, t_{i+1}), \\ (\texttt{deny}, \perp) & \text{if } risk_C(u,p) \geqslant t_n. \end{cases}$$

4.4 The RBAC$_A$ Model

The appropriateness- and role-based access control (or RBAC$_A$) model augments the standard RBAC96 model with two functions $\gamma : P \times R \to (0, 1]$ and $\lambda : P \to M$, where $\gamma(p, r)$ denotes the degree of appropriateness with which p is assigned to r, and $\lambda(p)$ denotes the risk mitigation strategy associated with p's usage. Similarly, for all $(p, r) \in PA$, we require $\gamma(p, r) \neq 0$.

Like RBAC$_C$, RBAC$_A$ introduces a similar approach to computing the risk of granting requests, although the notion of appropriateness is defined on the permission-role assignments. Given (u, p), we write $*p$ to denote the set of roles to which p is explicitly assigned. We write $*p \cap \downarrow u$ for the set of roles in $*p$ for which u is authorized. In other words, $*p \cap \downarrow u$ is the set of roles that are explicitly authorized for p and that lie on an au-path between u and p.

Given $p \in P$, p might be explicitly assigned to multiple roles, and each of these assignments is associated with a certain degree of appropriateness. A user u can

use p by activating the most appropriate role to which p is assigned, and certainly this role is the one that is considered when evaluating the risk of granting the access request. Hence, we define the risk function $risk_A : U \times P \to [0, 1]$ to be

$$risk_A(u, p) = \begin{cases} 1 & \text{if } *p \cap \downarrow u = \emptyset, \\ 1 - \max\{\gamma(p, r) : r \in *p \cap \downarrow u\} & \text{otherwise.} \end{cases}$$

Take an example of an RBAC$_A$ state in Fig. 1(b). We can see that u_2 is able to perform p_1 through r_1 or r_2. However, the role r_1 is the most appropriate one to which p_1 is assigned, therefore, the γ value of $\frac{1}{2}$ could be taken, and $risk_A(u_2, p_1) = \frac{1}{2}$.

Given an RBAC$_A$ state (V, E, γ, λ), an access request (u, p) and a risk mitigation strategy $\lambda(p)$ for p, the authorization decision function $Auth_A$ is defined as:

$$Auth_A((V, E, \gamma, \lambda), (u, p), \lambda(p)) = \begin{cases} (\texttt{allow}, \bot) & \text{if } risk_A(u, p) < t_1, \\ (\texttt{allow}, b_i) & \text{if } risk_A(u, p) \in [t_i, t_{i+1}), \\ (\texttt{deny}, \bot) & \text{if } risk_A(u, p) \geqslant t_n. \end{cases}$$

5 A Risk-Aware RBAC Model

A risk-aware RBAC model may combine the features of two or more of the RBAC$_T$, RBAC$_C$ and RBAC$_A$ models. For the sake of completeness, we consider the risk-aware RBAC (or R^{2}BAC) model that supports all the features of the RBAC$_T$, RBAC$_C$ and RBAC$_A$ models. In other words, we now work with the directed, labeled graph $G = (V, E, \alpha, \beta, \gamma, \lambda)$.

As in RBAC$_C$ and RBAC$_A$, to compute the risk of granting a request (u, p) in G, we firstly need to decide how to compute the risk associated with an au-path from u to p based on (α, β, and γ). Unlike our simpler models, we must then decide how to combine the risk of all au-paths between u and p into an appropriate risk value. We believe that there are at least two approaches to computing the risk associated with an au-path from u to p.

Given an au-path $u, r, \ldots, r', p$, one possibility is to define the risk associated with this path to be

$$1 - \min\{\alpha(u), \beta(u, r), \gamma(r', p)\}.$$

In other words, the risk of the au-path $u, r, \ldots, r', p$ is determined by the minimum value in the set comprising u's trustworthiness, the degree of competence for u to perform r, and the degree of appropriateness for p to be assigned to r'. Intuitively, an untrustworthy user still has a high likelihood of misusing her granted permission, even if she can invoke the permission through a role for which she is entirely competent and for which the permission is entirely appropriate. Similarly, a trustworthy user still has a high likelihood of misusing a permission

if she can only perform the permission through a role for which she has little competence or for which the role is rather inappropriate for the permission.

An alternative way of computing the risk associated with a path is to compute

$$\min\{1, (1-\alpha(u)) + (1-\beta(u,r)) + (1-\gamma(r',p))\}.$$

This computation acknowledges that there are risks associated with each part of the path and accumulates those risks.

Of course, it may be appropriate to compute the risk associated with a path as a more complex function of $\alpha(u)$, $\beta(u,r)$ and $\gamma(r',p)$. We defer the exploration of this matter, which would require substantial experimental validation, to future work. The purpose of this paper is to provide a risk-aware RBAC model that will provide a robust framework for the investigation of these issues.

We now consider how to combine the risks associated with multiple paths. Given $u, p \in V$, let $\Pi(u,p)$ denote the set of au-paths between u and p, and for each $\pi \in \Pi(u,p)$, let $risk(\pi)$ denote the risk associated with au-path π. We define $risk : U \times P \to [0,1]$, where

$$risk(u,p) = \begin{cases} 1 & \text{if } \Pi(u,p) = \emptyset, \\ \min\{risk(\pi) : \pi \in \Pi(u,p)\} & \text{otherwise.} \end{cases}$$

Note that, as in RBAC_C and RBAC_A, the way of computing the risk of allowing u to perform p in R^2BAC is to choose the minimum value from the risk of all au-paths between u and p.

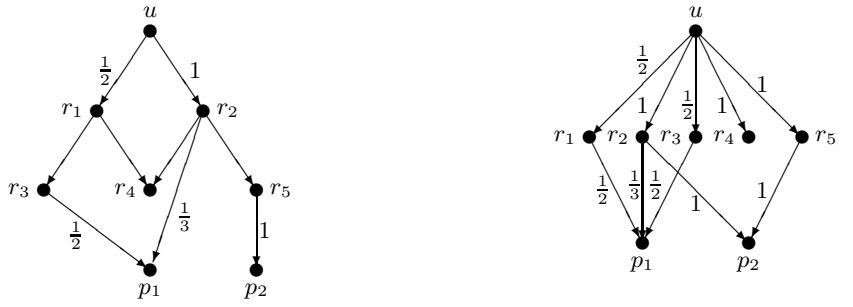
Consider the directed graph of an R^2BAC configuration shown in Fig. 2(a), where β and γ values are indicated by labels attached to edges, and assume that $\alpha(u) = 1$. There exist two au-paths from u to p_1, that is $u \to r_1 \to r_3 \to p_1$ and $u \to r_2 \to p_1$. If we use the first approach to compute the risk of those two au-paths, then $risk(u, r_1, r_3, p_1) = 1 - \frac{1}{2} = \frac{1}{2}$, and $risk(u, r_2, p_1) = 1 - \frac{1}{3} = \frac{2}{3}$. Therefore, the risk of granting u to perform p_1 is determined by the risk associated with the au-path $u \to r_1 \to r_3 \to p_1$, that is, $risk(u, p_1) = \frac{1}{2}$. If we use the second approach to compute the risk of those two au-paths, then $risk(u, r_1, r_3, p_1) = 1$, and $risk(u, r_2, p_1) = \frac{2}{3}$. Hence the risk of granting (u, p_1) is $\frac{2}{3}$, which is determined by the risk associated with the au-path $u \to r_2 \to p_1$.

Given an R^2BAC state $G = (V, E, \alpha, \beta, \gamma, \lambda)$, an access request (u, p) and a risk mitigation strategy $\lambda(p)$ for p, an authorization decision function $Auth$ is defined in the same way as before, that is:

$$Auth((V,E,\alpha,\beta,\gamma,\lambda),(u,p),\lambda(p)) = \begin{cases} (\texttt{allow}, \perp) & \text{if } risk(u,p) < t_1, \\ (\texttt{allow}, b_i) & \text{if } risk(u,p) \in [t_i, t_{i+1}), \\ (\texttt{deny}, \perp) & \text{if } risk(u,p) \geqslant t_n. \end{cases}$$

5.1 On the Advantages of Flat Risk-Aware RBAC

Given an R^2BAC state $G = (V, E, \alpha, \beta, \gamma, \lambda)$ and a request (u, p), we could use a breadth-first search algorithm to find all au-paths between u and p, and then



(a) Hierarchical R^{2}BAC state (b) Equivalent flat R^{2}BAC state

Fig. 2. A graphical representation of R^{2}BAC states

apply the risk function $risk(u, p)$ on all auth-paths to obtain a risk value. In many applications, we require a rapid response from the access decision function. In a risk-aware, hierarchical RBAC system with many users and permissions, this decision function, which depends on the computation of risk for multiple paths, may have unacceptably high overheads.

Based on this observation we discuss one way in which these performance issues might be addressed. In particular, we transform a hierarchical R^{2}BAC state into an equivalent flat R^{2}BAC state $G' = (V, E', \alpha, \beta', \gamma', \lambda)$ (in the sense that the risk of each request remains the same and, therefore, the authorization decision function returns the same decision) using the following procedure.

1. For all $u \in U$ and for all $r \in \downarrow u \cap R$, we define $(u, r) \in UA'$;
2. For all $p \in P$ and for all $r \in \uparrow p \cap R$, we define $(p, r) \in PA'$;
3. For all $(u, r) \in UA'$, define $\beta'(u, r) = \max\{\beta(u, r') : (u, r') \in UA, r' \geqslant r\}$;
4. For all $(p, r) \in PA'$, define $\gamma'(p, r) = \max\{\gamma(p, r') : (p, r') \in PA, r' \leqslant r\}$;
5. Define $E' = UA' \cup PA'$.

In the first two steps, we make explicit the user- and permission-role assignments that were previously implied by the role hierarchy. In the next four steps we ensure that each user- and permission-role assignment is associated with an appropriate β or γ value.

We now show how the transformation works by taking the example of the R^{2}BAC state illustrated in Fig. 2(a). The first two steps remove the need for the role hierarchy by explicitly assigning u to roles r_3, r_4 and r_5, the permission p_1 to roles r_1 and r_2 and p_2 to r_2. Then $\beta'(u, r_4)$ takes the maximum of the two values 1 and $\frac{1}{2}$, so $\beta'(u, r_4) = 1$ using Step 3. Similarly, Step 4 allows us to compute γ values for p_1 and p_2. Finally, we output the flat R^{2}BAC state shown in Fig. 2(b).

Having "flattened" our hierarchical RBAC state, all au-paths have length 2. We can then use our chosen method for computing the risk associated with an au-path to compute the risk associated with a request, as before. We can now

use this risk value as part of the input to our risk-aware authorization decision function.

Of course, the likely reduction in time taken to decide access requests is offset by the fact that greater storage is required for the RBAC relations. Perhaps more important, however, is the increased difficulty in ensuring consistent updates to the RBAC state: one of the great virtues of hierarchical RBAC is that it simplifies the management of user- and permission-role assignments, since many such assignments are implied by the role hierarchy. It is likely that different trade-offs will be tolerated for different applications and different contexts. An investigation of these trade-offs would be an interesting subject for future work.

5.2 On Sessions in Risk-Aware RBAC

The RBAC96 model introduces the notion of *sessions* to achieve the principle of *least privilege* [22] in RBAC systems. Until this point we have ignored sessions, which are an important part of the RBAC96 model.

A user may create one or more sessions: in each session the user only activates a set of roles that are required to accomplish her task. Conceptually, a session is associated with a user and is a subset of the roles for which a user is authorized. A request (u, p) is authorized if there exists a session s associated with u and a role r in s such that there is an au-path from r to p.

In terms of our graph-based formalism, we may introduce a new graph $G_{\text{Dyn}} = (V_{\text{Dyn}}, E_{\text{Dyn}})$ to represent the run-time state. This graph is derived from the RBAC96 graph in the following way. Writing S to denote the set of sessions, $V_{\text{Dyn}} = S \cup R \cup P$ and $(s, r) \in E_{\text{Dyn}}$ if role r has been activated in session s. A request is now modeled as a pair (s, p),[3] where s is a session, and is authorized only if there exists an au-path from s to p.

We can very easily extend the above approach to our risk-aware formalism. Specifically, risk is calculated over paths in G_{Dyn}, rather than G. In other words, the risk computation now applies to a session, rather than a user, so the end-user may find different mitigations being applied, depending on the session she chooses to activate.

6 Related Work

There has been significant research on risk-aware access control for enabling the secure sharing of information within or across multiple organizations [7,8,14,16,18,24,25]. However, most works attempt to achieve this goal by proposing approaches based on risk estimation and economic mechanisms [7,14,16,18,24]. On the other hand, there are only a few papers on extending RBAC models with risk semantics [2,5,11,19]. Unlike our models, none of them is concerned with the *authorization* semantics of risk-aware RBAC models. In this section, we review some work that are most related to ours, and illustrate the importance of our risk-aware RBAC models.

[3] A session is analogous to a subject in models based on a protection matrix.

Cheng *et al* [7] recently introduced a Fuzzy MLS system that controls the user's read access to information. The risk of granting a user to read a data object is estimated based on the security label of the object and the degree of the trustworthiness of the user. A number of explicit formulae is provided to compute the trustworthiness of the user based on the security clearance and category set of the user. On the other hand, Srivatsa *et al* [24] proposed a trust and key management paradigm for securing information flows across organizations, where the trustworthiness of a user is computed using a dynamic trust metric that depends on user's behavior. We believe that these approaches to computing trustworthiness of users can be used to specify the α function in our risk-aware RBAC models.

In addition, Cheng *et al* [7] suggested a global approach to risk management in the Fuzzy MLS model. They define a "hard" boundary, above which all accesses are denied, and a "soft" boundary, below which all accesses are allowed. Between the hard and the soft boundaries, an access request is allowed only if a risk mitigation mechanism can be applied to the access. In our work, we apply similar techniques to RBAC. However, we adopt a more sophisticated treatment of risk mitigation.

There has also been some work on incorporating risk semantics in the RBAC model. Nissanke and Khayat [19] assumed the existence of a partially ordered set of risk levels, and assigned these risk levels to permissions in an RBAC system. Therefore, the usage of one permission might be more risky than the other according to the risk levels to which these permissions are assigned. They also suggested an approach to reorganize the role hierarchy using risk analysis of permissions. In contrast to their work, we are concerned with how much risk will be incurred by allowing users to perform permissions, and provide explicit methods of computing such risk. Dimmock *et al* [11] extended the OASIS RBAC model [3] to make decisions on the basis of trustworthiness of users and cost of actions for certain outcomes. Unlike our models, the extended OASIS model returns binary decisions: a user's request to take an action with certain outcome is denied if the trustworthiness of the user is too low for the outcome's cost, and allowed otherwise. Furthermore, Aziz *et al* [2] introduced a refined RBAC model with the consideration of risk associated with operational semantics of permissions, collective usage of permissions, and conflicts of interest respectively. Celiker *et al* [5] introduced a probabilistic risk management framework to measure and evaluate users' risk in an RBAC system. Unlike our models, neither of these approaches support a risk-aware evaluation mechanism that is able to return richer types of access control decisions.

In summary, although all above work attempted to study risk in the context of RBAC, none of them has considered the possible ways of quantifying risk in the components of the RBAC model, and examined the way of extending the access control decision function in RBAC to become risk-aware.

7 Concluding Remarks

In this paper we have examined a number of possible ways to define risk in different components of the RBAC model. In particular, we observed that the risk of granting a request in the RBAC model could be rephrased in terms of user trustworthiness, the degree of competence of a user-role assignment, or the degree of appropriateness of a permission-role assignment. We assume that there exist appropriate software components that are able to evaluate these factors, and dynamically adjust the degree of these factors when context is changed.

Moreover, we developed three simple risk-aware RBAC models that consider those three quantitative factors respectively. We used a graph-based formalism of RBAC96 as a basis for defining the semantics of these models, and suggested the association of risk mitigation strategies with permissions. The resulting models have clear authorization semantics and accommodate the awareness of risk in deciding access requests.

Finally, we proposed a full risk-aware RBAC model that combines all the features of three simple models, and considered some of the practical issues that might arise when implementing such a model. To our knowledge, this is the first model that defines quantitative factors on various components of the RBAC model, and studies the way of combining these factors in order to acquire an appropriate method of computing the risk of allowing access.

There are several interesting directions for future work. As described above, we introduced the β and γ functions to quantify the competence of user-role assignments and appropriateness of permission-role assignments respectively. However, we did not provide an explicit way of specifying these two functions. We are currently working on an approach that constructs β and γ from the structure of the RBAC96 graph. In particular, we propose a formula for β, for example, that is able to compute the competence values for all user-role relations including those that are not encoded in the RBAC96 graph.

One interesting possibility for future work is to develop context- and risk-aware RBAC models. In particular, we would like to define a matrix of risk mitigation strategies to be associated with each permission, where each row represents a different context. The context could be as simple as emergency or non-emergency situations (as used in break-glass policies [4]) or we could monitor whether there are alternative more senior users available to invoke the permission (as used in the auto-delegation mechanism [10]).

We also would like to extend our models to include user obligations [13], and use the idea of "charging for risk" to enforce those obligations. The risk charge is removed from the user's "risk account" if the user fulfils the obligation. However, if the obligation is not fulfilled, the risk charge increases the risk associated with any subsequent requests made by the user. A user who is unable or unwilling to fulfil her obligations will eventually be denied all access requests. In other words, the honest user has an incentive to fulfil her obligations.

We would also like to extend our model to include usage control [21]. Of particular interest would be the way in which obligations might be used as a feedback mechanism to modify risk mitigation strategies themselves. In this way,

risk-awareness and risk mitigation might become responsive to previous access requests and system activity.

Finally, we hope to develop a metamodel that captures the different ways of interactions between authorization and obligation. We could then construct a series of role-based models that instantiate one or more features of the metamodel. In particular, the metamodel enables us to develop new risk-aware RBAC models that provide more flexible and sophisticate ways of associating obligations with risk-aware authorization decisions.

Acknowledgements. We would like to thank the anonymous reviewers for their helpful feedback.

This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. American National Standards Institute: American National Standard for Information Technology – Role Based Access Control (2004), ANSI INCITS 359-2004
2. Aziz, B., Foley, S.N., Herbert, J., Swart, G.: Reconfiguring role based access control policies using risk semantics. Journal of High Speed Networks 15(3), 261–273 (2006)
3. Bacon, J., Moody, K., Yao, W.: A model of OASIS role-based access control and its support for active security. ACM Transactions on Information and System Security 5(4), 492–540 (2002)
4. Brucker, A.D., Petritsch, H.: Extending access control models with break-glass. In: Proceedings of the 14th ACM Symposium on Access Control Models and Technologies, pp. 197–206 (2009)
5. Celikel, E., Kantarcioglu, M., Thuraisingham, B.M., Bertino, E.: A risk management approach to RBAC. Risk and Decision Analysis 1(1), 21–33 (2009)
6. Chen, L., Crampton, J.: On spatio-temporal constraints and inheritance in role-based access control. In: Proceedings of the 2008 ACM Symposium on Information Computer and Communications Security, pp. 356–369 (2008)
7. Cheng, P.C., Rohatgi, P., Keser, C., Karger, P.A., Wagner, G.M., Reninger, A.S.: Fuzzy multi-level security: An experiment on quantified risk-adaptive access control. In: Proceedings of the 2007 IEEE Symposium on Security and Privacy, pp. 222–230 (2007)
8. Clark, J.A., Tapiador, J.E., McDermid, J.A., Cheng, P.C., Agrawal, D., Ivanic, N., Slogget, D.: Risk based access control with uncertain and time-dependent sensitivity. In: Proceedings of the International Conference on Security and Cryptography, pp. 5–13 (2010)
9. Crampton, J., Huth, M.: Detecting and countering insider threats: Can policy-based access control help? In: Proceedings of the 5th International Workshop on Security and Trust Management (2009)

10. Crampton, J., Morisset, C.: An Auto-Delegation Mechanism for Access Control Systems. In: Cuellar, J., Lopez, J., Barthe, G., Pretschner, A. (eds.) STM 2010. LNCS, vol. 6710, pp. 1–16. Springer, Heidelberg (2011)
11. Dimmock, N., Belokosztolszki, A., Eyers, D.M., Bacon, J., Moody, K.: Using trust and risk in role-based access control policies. In: Proceedings of the 9th ACM Symposium on Access Control Models and Technologies, pp. 156–162 (2004)
12. Ferraiolo, D.F., Kuhn, D.R.: Role-based access controls. In: Proceedings of the 15th National Computer Security Conference, pp. 554–563 (1992)
13. Irwin, K., Yu, T., Winsborough, W.H.: On the modeling and analysis of obligations. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 134–143 (2006)
14. JASON Program Office: Horizontal integration: Broader access models for realizing information dominance. Technical Report JSR-04-132, MITRE Corporation (2004)
15. Landoll, D.J.: The Security Risk Assessment Handbook: A Complete Guide for Peforming Security Risk Assessments. CRC Press (2005)
16. Molloy, I., Cheng, P.C., Rohatgi, P.: Trading in risk: Using markets to improve access control. In: Proceedings of the 2008 Workshop on New Security Paradigms, pp. 107–125 (2008)
17. National Institute of Standards and Technology: Risk Management Guide for Information Technology Systems (2002), NIST Special Publication 800-30
18. Ni, Q., Bertino, E., Lobo, J.: Risk-based access control systems built on fuzzy inferences. In: Proceedings of the 5th ACM Symposium on Information Computer and Communications Security, pp. 250–260 (2010)
19. Nissanke, N., Khayat, E.J.: Risk based security analysis of permissions in RBAC. In: Proceedings of the 2nd International Workshop on Security in Information Systems, pp. 332–341 (2004)
20. Moses, T. (ed.): OASIS: eXtensible Access Control Markup Language (XACML) Version 2.0, OASIS Standard (February 1, 2005)
21. Park, J., Sandhu, R.S.: The UCON_{ABC} usage control model. ACM Transactions on Information and System Security 7(1), 128–174 (2004)
22. Saltzer, J.H., Schroeder, M.D.: The protection of information in computer systems. Proceeding of the IEEE 63(9), 1278–1308 (1975)
23. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. IEEE Computer 29(2), 38–47 (1996)
24. Srivatsa, M., Balfe, S., Paterson, K.G., Rohatgi, P.: Trust management for secure information flows. In: Proceedings of the 15th ACM Conference on Computer and Communications Security, pp. 175–188 (2008)
25. Zhang, L., Brodsky, A., Jajodia, S.: Toward information sharing: Benefit and risk access control (BARAC). In: Proceedings of the 7th IEEE International Workshop on Policies for Distributed Systems and Networks, pp. 45–53 (2006)