# Extending RBAC for Large Enterprises and Its Quantitative Risk Evaluation

**Seiichi Kondo**[1], **Mizuho Iwaihara**[2], **Masatoshi Yoshikawa**[2], **Masashi Torato**[3]

[1]Mitsubishi Electric Corporation, Kamakura-city, Kanagawa, Japan
Kondo.Seiichi@dr.MitsubishiElectric.co.jp
[2]Kyoto University, Sakyo-ku, Kyoto, Japan
{iwaihara, yoshikawa}@i.kyoto-u.ac.jp
[3]Mitsubishi Electric Information System, Kamakura, Kanagawa, Japan
torato-masashi@mdis.co.jp

**Abstract.** Systems and security products based on the RBAC model have been widely introduced to enterprises. Especially, the demands on enforcement of enterprise-level security policies and total identity management are rapidly growing. The RBAC model needs to be extended to deal with various circumstances of large enterprises, such as geographical distribution and heterogeneous environments including physical access control. In this paper, we introduce a new RBAC model, suitable for single sign-on systems. This model optimizes evaluation of rule-based RBAC so that total operation costs and productivity can be improved.

Furthermore, to select most cost-effective RBAC extensions for enterprise-wide requirements, we propose a quantitative risk evaluation method based on fault trees. We construct fault trees having security violation and productivity loss as top events, and RBAC standard functions and security incidents as basic events. Probabilities of the top events are computed for given RBAC models and operation environments. We apply this method to a real enterprise system using the above RBAC extension and the proposed model realizes more safety and productivity over the base model.

*Please use the following format when citing this chapter:*

Kondo, S., Iwaihara, M., Yoshikawa, M., Torato, M., 2008, in IFIP International Federation for Information Processing, Volume 286, Towards Sustainable Society on Ubiquitous Networks, eds. Oya, M., Uda, R., Yasunobu, C., (Boston: Springer), pp. 99 - 112.

# 1 Introduction

Total and centralized management of identities and access control of large enterprises has been in focus, due to the increasing demands on corporate governance over information securities. Access control policies based on the role-based access control (RBAC) model [1][2][3] are becoming widely accepted and deployed to corporate information systems. However, as the RBAC is applied to many situations, we encounter a number of new challenges. We need to deal with geographically distributed, heterogeneous security objects, while consistencies of access control information have to be maintained in such a distributed environment. We need to find solutions which take into account the tradeoffs between total cost of system development, operations, and threat-levels of security events.

In a distributed enterprise-wide system, the task of provisioning changes on access control information such as roles and privileges becomes a big challenge, where changes are caused by various reasons, such as hiring, retirement and reallocation of personnel, installation of new facilities, and amendment of security policies. However, since there are various practical constraints in computing and distributing such changes, delays in reflecting the changes are inevitable. Delays in authorizing privileges will lead to productivity loss, while delays in revocation of privileges will lead to the risk of unauthorized access by non-admitted users. We propose a new extended RBAC model which can prevent or reduce such risks. It can be used for providing access control information efficiently by utilizing organizational and position information expressed in rules, so that changes on privileges are computed at authorization time instead of the time changes occur, having the effect of reducing provisioning cost.

The original RBAC model has been extended in various directions to deal with new requirements and target systems, such as Enterprise RBAC [4][5], rule-based approach [6][7][8], including this paper. However, it is hard to say a particular model fits given requirements with admissible costs, because there is not an established way of comparing RBAC extensions in terms of safety and cost effectiveness. Extended RBAC models have been compared with existing ones by qualitative analysis or by evaluation on a particular implementation [9]. In this paper, we propose a systematic and qualitative risk evaluation method for RBAC models, consisting of the following elements: (1) Common fault trees [14][15] are constructed based on the core functions of RBAC, augmented with functions for enterprise-level RBAC, and risk events obtained from past accidents and incidents[11][12]13]. (2) Subtrees necessary for evaluation are selected from the common fault trees systematically constructed at (1), and logic programs are generated from the AND-OR constructs of the fault trees. (3) Risk values are computed from the program of (2) by entering basic parameters to the program, where some parameters are obtained by measuring performance of a particular target system, while some parameters are obtained from empirical knowledge and surveys through questionnaires [11][13]. Using this evaluation method, we can quantitatively estimate in which circumstance our proposed model can prevent or reduce risks quantitatively.

The rest of this paper is organized as follows: In Section 2, we survey technologies related to management of access control information. In Section 3, we present quantitative risk evaluation method for RBAC and its extensions. In Section 4, we

present a RBAC extension for large enterprises and apply the quantitative evaluation

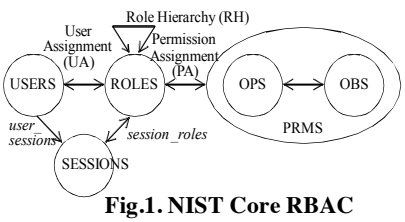


**Fig.1. NIST Core RBAC**

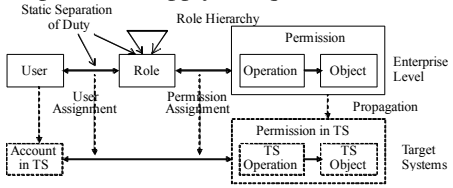


**Fig.2. Enterprise RBAC Model**

method of Section 3. Section 5 is a conclusion.

## 2 Related Technologies

In this section, we show basic definitions of RBAC models and related concepts to be used for our risk analysis.

### 2.1 Role-Based Access Control Model and its Variations

RBAC model [1][2][3] maintains the mapping between users and security objects via roles, instead of maintaining a direct mapping between them. This enables updates of user attributes such as organizations and positions, and security objects such as files and business applications, independently from each other. Fig.1 shows the standard model of the Hierarchical RBAC of NIST(National Institute of Standards and Technology)[2]. The following is the definition of the Core RBAC.
Definition1: Core RBAC

- $USERS, ROLES, OPS, and\ OBS$, users, roles, operations and objects, respectively.
- $UA \subseteq USERS \times ROLES$, a many-to-many mapping user-to-role assignment relation.
- $assigned\_users(r) = \{u \in USERS \mid (u, r) \in UA\}$, the mapping of role $r$ onto a set of users.
- $PRMS = 2^{(OPS \times OBS)}$, the set of permissions.
- $PA \subseteq PRMS \times ROLES$, a many-to-many mapping permission-to-role assignment relation.
- $assigned\_permissions(r) = \{p \in PRMS \mid (p, r) \in PA\}$, the mapping of role $r$ onto a set of permissions.
- $SESSIONS$, the set of sessions.
- $user\_sessions(u : USERS) \rightarrow 2^{SESSIONS}$, the mapping of user $u$ onto a set of sessions.
- $session\_roles(s_i) \subseteq \{r \in ROLES \mid (session\_users(s_i), r) \in UA\}$, the mapping of session $s$ onto a set of roles.
- $avail\_session\_perms(s : SESSIONS) \rightarrow 2^{PRMS}$, the permissions available to a user in a sessions.

The Core RBAC has the following Administrative Commands and System Functions, for managing and operating on the above constructs.
(1) Administrative Commands for Core RBAC

```
-AddUser(user:NAME)
-DeleteUser(user:Name)
-AddRole(role:NAME)
-DeleteRole(role:NAME)
-AssignUser(user, role:NAME)
-DeassignUser(user, role:NAME)
-GrantPermission(object, operation, role:NAME)
-RevokePermission(operation, object, role:NAME)
```

(2) System Functions for Core RBAC

```
-CreateSession(user:NAME; ars:2^NAMES; session:NAME)
-DeleteSession(user, session:NAME)
-AddActiveRole(user, session, role:NAME)
-DropActiveRole(user, session, role:NAME)
-CheckAccess(session, operation, object:NAME; out result:BOOLEAN)
```

Enterprise Role-Based Access Control (ERBAC) Model [4][5] has been proposed for user and access privilege management over all the systems of enterprise IT environment. The model can deal with multiple target systems, and its roles consist of combinations of diverse and system-specific privileges. Fig.2 depicts the ERBAC Model. We extend ERBAC for modeling provision of access control information. The extended ERBAC becomes the common underlying model for FTA (Fault Tree Analysis).

In the enterprise level, user assignments are usually defined using rules. In [6], the rule-based RBAC is shown, where rules defined over user attributes are evaluated at runtime to grant accesses. In this model, changes to attributes of users do not invoke changes to the user assignment, thus reducing operation costs of such changes, which are frequent in personnel changes. In [7], the Rule-Based Provisioning of RBAC is proposed, where dynamic rule-based assignment is used at the enterprise level, while static assignment is used at the target systems. In [8], a rule-based framework is proposed for role-based delegation and revocation. In [9], systematic control and management architecture of data integrity based on metadata management is shown.

## 2.2 Quantifying Costs and Risks of Security Policies

One of the reasons why there has been significant delay in adopting current information security measures to cooperate information systems is that the risks of IT security incidents are unclear, and thus it is hard to determine proper amount of investment. Therefore, quantitative assessment of security risks is in great demand. Risk quantification methods have been studied in several industrial sectors. In the banking sector, Basel Accord of 2004 (Basel II) [10] requires maintenance of regulatory capital reflecting three major components of risk that a bank faces: credit risk, operational risk and market risk. Basel II operational risk, which is newly added, includes the following event types related to information systems: internal/external frauds, software/hardware failures, and data entry errors.

The study of the economic impact of RBAC [11] gives quantitative definition of Operating Benefits $OB_{it}$ per employee as follows:

i indexes industry, and t indexes year.

$OB_{it} = AC_{it} + PB_{it} + SB_{it}$

$OB_{it}$ = operating benefits per employee

$AC_{it}$ = administrative cost reductions per employee
$PB_{it}$ = productivity benefits per employee
$SB_{it}$ = security benefits per employee
The following three observations are reported as end-user benefits of RBAC:
(1) RBAC reduces administrative processing time.
(2) RBAC increases productivity.
(3) RBAC reduces the frequency and severity of security violations.

In (3), [12][13] are referred for the results of crime and security survey. In deriving (1), the following activities are quantitatively compared between RBAC and non-RBAC:

- Assigning existing privileges to new users,
- Changing existing users' privileges,
- Establishing new privileges for existing users, and
- Terminating privileges.

In order to operate these activities, delay time occurs from getting privileges to enabling them on real systems. The effect of downtime reduction is hard to be directly compared with (1), because if an employee is added, downtime for (2) is from the point s/he is added and to the point privileges are given to her/him. On the other hand, downtime for (1) is from the point when the administrator received change information and to the point the administrator reflects the change to the system. In this paper, we aim at measuring the effect to the system, so that we regard both (1) and (2) are productivity loss for employees. For (3), it is pointed out that the effort for reflecting security policies to the system is reduced as a benefit of introducing RBAC. This implies that the processing time reduction of (1) contributes to the reduction of time interval where access control states deviate from the security policies. For example, by reducing the time for terminating privileges, the probability of unauthorized access through the deviated access control states can be reduced.

## 2.3 Fault Trees for Security System Design and Analysis

FTA (Fault Tree Analysis)[14] has been utilized for safety-critical systems such as nuclear power plants, aircrafts, and artificial satellites. Recently, applications of FTA to analysis on failures and security of information systems are reported. In [15], design analysis of security-critical systems utilizing FTA is discussed.

Once a fault tree is constructed, the probability of the top event can be calculated as described in Fault Tree Handbook [14]. Thus construction of fault trees is important. In [15], it is pointed out that distinction of a system and components, and refinement of diversified security concerns are important issues.

The 12th Annual Computer Crime and Security survey [13] presents a list of twenty types of threats, including "Insider abuse of Net access," "Virus," "Laptop/mobile device theft". In this paper, we show construction of fault trees having core functions of RBAC as basic components, and also security violation and productivity loss as top events, and propose a framework for quantitatively analyzing cost effectiveness of RBAC variations, utilizing these fault trees.

# 3 Quantitative Analysis of Effectiveness of Applied RBAC Models

## 3.1 Overall Structure of Analysis

To incorporate access control policies based on an extended RBAC model, every functionality of the system has to meet the following requirements:

(1) Users unauthorized by the access control policies should not be able to execute requesting privileges.
(2) Users authorized by the access control policies should be able to execute requesting privileges.

The system may momentarily fall into invalid states due to significant events such as staff reallocation and accidents, and certain costs are expected to bring the system back into valid states. We try to measure these costs and the risks raised by the invalid states.

Faults regarding securities can be classified into direct and indirect faults. Direct faults are those which actually cause damage, while indirect faults are those which may not cause damage but cause violation of security rules which are intended to prevent such damage and imposed by the organization. If the system faces internal or external attacks while the settings of the system are violating company policies, damages are likely. Namely, the probabilities of incidents or accidents are strongly influenced by the probabilities of breach of company policies, which are in turn determined by the duration of exposure to dangerous states per unit time. The reduction of administrative processing time can contribute to improvement of securities. Therefore we use the probabilities of policy violation per unit time as basic components. For example, erroneous settings of access privileges and delayed effect of new settings due to slow processing time can be considered as basic components.

## 3.2 Basic Steps of Analysis

We propose a method for quantitatively evaluating given RBAC variations applied to given target systems. It proceeds by the following four steps:

(1) Constructing fault trees common to RBAC models: Defining fault trees having "security violation" and "productivity loss" as top events, and functions of the RBAC model and its variations as intermediate and basic events.
(2) Constructing model-specific fault trees: Selecting events relevant to the model under evaluation from the common fault trees of (1).
(3) Determining system and performance parameters: Event probabilities of basic and intermediate components are obtained from characteristics of the target system. In our model, probabilities are proportional to the ratio of elapsed time of basic and intermediate events to a unit time, such as a day.
(4) Implementing risk evaluation functions: The results of (1)(2)(3) are compiled into logic formulae or programs, and then evaluation results can be computed.

## 3.3 Target Access Control Systems

We consider functions of RBAC necessary for enterprise-level integrated access control systems, consisting of the following components:

(1) Enterprise-level Access Control System: administrates identity and access control information, observing enterprise-level security policies.
(2) Target Systems: execute application-specific access control by using information provisioned by the Enterprise-level Access Control System.
(3) Authorization Information Provider: provisions identity and access control information to target systems in the methods the target systems require. We need to be able to model two types of provisioning: batched prior provisioning and real-time change-triggered provisioning.

In the following, we discuss time cost in provisioning change information from the enterprise-level integratedly-managed databases to the target system. We assume that the following information is managed in the system at whole:

(1) User identifiers.
(2) User attributes (passwords, smart card information, certificates, etc)
(3) User groups (a group is a construct structuring users, corresponding to organizations, projects, qualifications, and positions, et al.)
(4) Enterprise-level roles
(5) Enterprise-level UA
(6) Enterprise-level permissions
(7) Enterprise-level PA

Upon execution of an administrative command, its resulting changes shall be provisioned and synchronized with the target systems. We need to classify the tasks of computing each element to be provisioned and the tasks of provisioning according to the above (1)-(7), and execute evaluation on these tasks and the synchronization method adopted by the evaluating model. Performance and cost evaluations are influenced by the ways of how changes on (1)-(7) are provisioned to the target systems. Changes may be provisioned in a separate or combined manner, and schedule-driven or change-driven manner. Fig.3 shows the case where the entire set of (1)-(7) is sent to the target system ts1. In Fig.3, when changes occur at the enterprise level, updated information of (1)-(7) is provisioned to ts1. UA(5) and PA(7) may be specified implicitly by rules implementing a policy. In this case, there are two choices: (a) rules

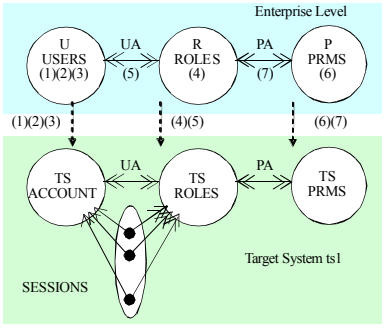


**Fig.3. Enterprise-Level Role Provisioning**

are evaluated to obtain concrete relationships between objects and the resultant relationships are provisioned, and (b) only rules are provisioned.

Upon comprehensive classification of the patterns of provisioning, we select patterns relevant to the operations of the evaluating model, and obtain the costs of the selected patterns. As an example, we consider User Addition as a frequent routine operation. It is executed by the following steps:

(1) Following the access policies, compute PRMS.
(2) Provision the user information.
(3) Provision permitted access control information.

We evaluate the performance cost of each step.

```
:- evaluate_user_provisioning(U_ID, U_Attr, APP_ID, add_user, Max_time,
Total_time).
evaluate_user_provisioning(U_ID, U_Attr, APP_ID, FuncName, Max_time,
Total_time) :-
    calculate_UA(U_ID, U_Attr, UA_List, UA_calculation_time),
    calculate_PA(UA_List, APP_ID, PA_List, PA_calculation_time),
    provision_U(APPID,FuncName,U_time),              % (1)(2)(3)
    provision_R(APPID,FuncName,R_time),              % (4)
    provision_P(APPID,FuncName,P_time),              % (6)
    provision_UA(APPID,FuncName,UA_List,UA_time),    % (5)
    provision_PA(APPID,FuncName,PA_List,PA_time),    % (7)
    max([U_time,R_time,P_time,UA_time,PA_time], Max_time),
    sum([U_time,R_time,P_time,UA_time,PA_time], Total_time).
% provision_U(in,in,out).
provision_U(ts1,add_user,(measured_users_provisioning_time)).
provision_R(ts1,add_user,0).
provision_P(ts1,add_user, 0).
provision_UA(ts1,add_user,UA_List,(measured_ua_provisioning_time)).
provision_PA(ts1,add_user,_,0).
```

## 3.4 Risk Analysis Utilizing Common RBAC Fault Trees

Now we describe security threat analysis by constructing fault trees common to the RBAC model and its variations. As we pointed out in Section 3, we list up the following two top events.

TE1 (Security violation): An unauthorized user executes an operation (unauthorized access) - surfaced risk.

TE2 (Productivity loss): An authorized user is unable to execute a permitted operation (loss of opportunity) – potential risk.

Following the above top events, we connect basic events Xi (i=1, ･･･, n) described below to the top events, and constructs AND-OR trees. Fig.4 shows fault trees we constructed for large-enterprise security systems based on RBAC and ERBAC. We selected basic risk events from: (1) risks introduced from [11][12][13], and (2) risks arising from provisioning processing time for the Administrative Commands and System Functions of [2], and time for computing provisioning information specific to the evaluating RBAC model as we discussed in Section 4.3. We selected as the processes of evaluating the rules regarding UA and PA, and basic information such as users, roles, and permissions as basic intermediate events of fault trees. We can break down access control risks and productivity losses into these operations by considering processing time for executing low-level operations which are different in each model.

Top level risks are systematically placed in the fault trees according to causalities. In TE1, we introduce the risks shown in CSI Computer Crime and Security Survey [13], and focus on the risks A111 and A112 which have direct influence on RBAC. Under the assumption that the probabilities of these events happening per unit time are uniform, securities can be reinforced by reducing the time interval where unnecessary privileges exist during execution of RBAC functions.

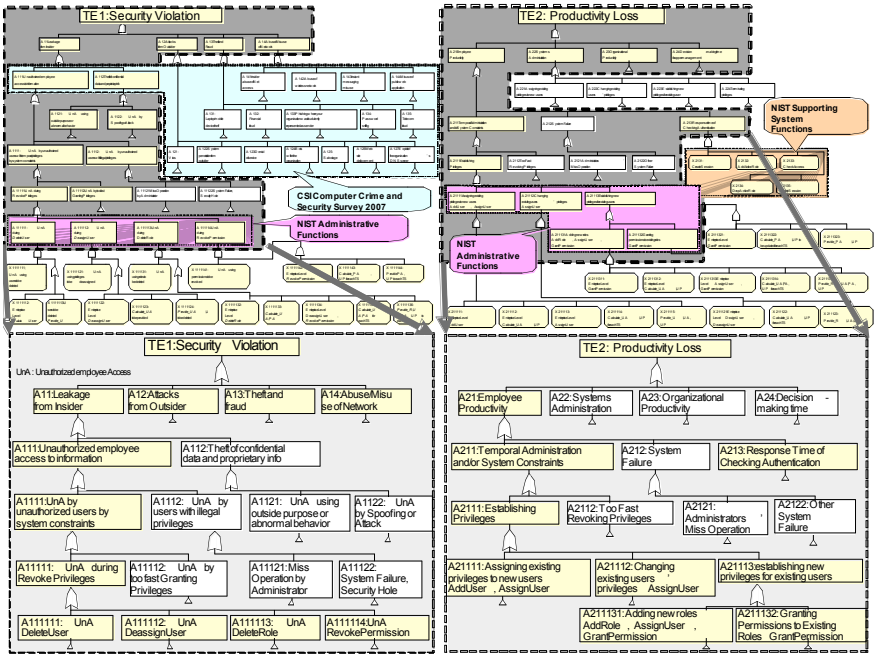


**Fig.4. Common RBAC Fault Trees**

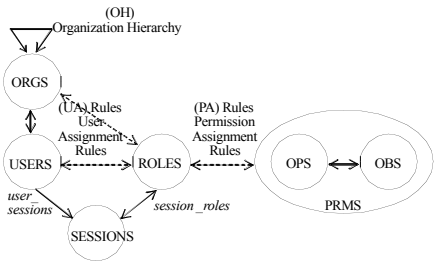


**Fig.5. Rule-Based Hierarchical Organization RBAC**

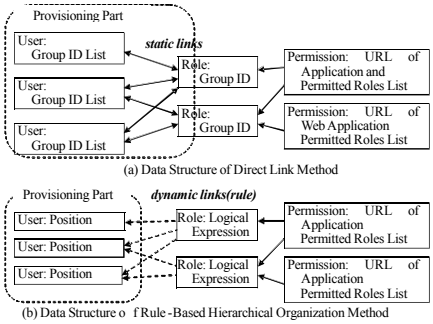


**Fig.6. Provisioning Part for Add_User**

# 4 Application to Rule-Based Hierarchical Organization RBAC and its Quantitative Evaluation

We consider the extension to the standard RBAC model, Rule-Based Hierarchical Organization RBAC for reducing batched prior provisioning costs. It is suitable for SSO (Single Sign On) of web applications. This model is implemented into real systems operating in enterprises of more than 50,000 employees, including manufacturing companies and financial companies. We apply the quantitative evaluation method of Section 3 to this system as case studies.

## 4.1 Model Definition

In web business applications, user information provisioning is often accompanied by access control information such as UA, whose provisioning becomes necessary when access control information is updated. To reduce provisioning costs, we propose an extension such that the organization is separated from the roles and mapping between them is computed locally from rules at authentication time as shown in Fig.5. We name as ts2 the target system based on this extended model.

In ts2, rules which implement policies are not evaluated during provisioning, but instead these rules are provisioned. Authorizations and permissions are determined by evaluating these rules at runtime. Therefore, the following is carried out for provisioning.

```
provision_U(ts2,add_user,(measured_users_provisioning_time)).
provision_UA(ts2,add_user,_,0).          % different from ts1
```

In the following, we define two methods, the first is without rule provisioning and the second is with rule provisioning. Fig.6 shows their data structures.

**Direct Link Method**: Security policies are pre-evaluated and then provisioned.

Tasks at access setting: `calculate_UA, provision_U, provision_UA`

Tasks at authorization: *`CheckAccess`*

**Rule-Based Hierarchical Organization Method**: Security policies are defined by rules, and only user information is provisioned. UA is computed at authentication and then permission is determined.

Tasks at access setting: `provision_U`

Tasks at authorization: `calculate_UA,` *`CheckAccess`*

*Model Definition of ts1 and ts2*

Apply A1111, A2111, and A213 of Fault Tree

```
:- evaluate_user_provisioning(user1, Attributes_list, ts1, add_user,
Time_of_Add_user_to_ts1). % A1111 and A2111 for ts1
:- evaluate_user_provisioning(user2, Attributes_list, ts2, add_user,
Time_of_Add_user_to_ts2). % A1111 and A2111 for ts2
:- check_authentication(ts1, check_on_demand_authorization,
  Time_of_check_authentication_1).   % A213 for ts1
:- check_authentication(ts2, check_on_demand_authorization,
 Time_of_check_authentication_2).    % A213 for ts2
:- check_authentication(ts1, check_authorization_on_authentication,
  Time_of_check_authentication_1).   % A213 for ts1
:- check_authentication(ts2, check_authorization_on_authentication,
```

**Table 1. Data structure for measurement.**

(a) Organization Structure

| | Number of Organization | Number of Users | Position |
|---|---|---|---|
| Level1 | 4 | 4 | Exective |
| Level2 | 6 | 48 | Senior Manager |
| Level3 | 150 | 450 | Manager |
| Level4 | 450 | 4,500 | Employee |
| 計 | 610 | 5,002 | |

(b) Objects, Permissions, and Rules

| | |
|---|---|
| No. of objects | 10 |
| No. of role per object | 2 |
| total | 16 |
| No. of UA Rule | 2 |
| No. of belonging groups | 10 |

**Table 2. Measurement results.**

| | On-demand authorization | | Authorization on authentication | |
|---|---|---|---|---|
| | Response Time (msec) | Throughput (transactions /sec) | Response Time (msec) | Throughput (transactions /sec) |
| Direct Link | | | | |
| single | 1.509 | 662.856 | 11.363 | 90.277 |
| 10 | 9.691 | 1,031.914 | 74.872 | 133.561 |
| Rule Based | | | | |
| single | 1.575 | 645.856 | 12.210 | 81.486 |
| 10 | 9.942 | 1,005.874 | 79.561 | 125.690 |

```
Time_of_check_authentication_2).    % A213 for ts2
```

## 4.2 Evaluation

The differences of the two models are illuminated by the above definitions. We therefore compare performances according to the differences. Parameters of the fault trees regarding processing time are sampled from the running systems.

Table 1 shows statistics of the benchmarked systems, where users are 5,000, and the role hierarchy has five levels. Roles are automatically generated for each organization and each position, and each object is assigned with 16 roles as permissible. Rules for User Assignment consist of two terms, because most of real system implementations we encounter have rules of one or two terms combined by AND or OR. The target system implements two different authorization methods described below, and we compared response time and throughput for each model.

***(1) On-demand authorization method***

Authorization decision is carried out at the timing when a request for activating a web application is received with the URL of the application and the requesting user. This type of authorization is often used for opening connections upon clicks on links of pages.

・*Direct Link Method:* Belonging group lists, which are obtained by policy evaluation, are provisioned to the ID management database. These belonging group lists are held during user authorization. Upon a request for activating a web application, (1-1) roles that are approved for activation of the application are retrieved, and (1-2) authorization is made by directly comparing the roles with the belonging group lists.

・*Rule Based Hierarchical Organization Method:* No link indicating UA is held in the ID management database, but instead UA is stored as logical expressions. During user authorization, belonging organizations and positions are held. Upon an activation request of a web application, (2-1) roles approved for activation of the application are retrieved, and then (2-2) UA expressions containing these roles as terms are obtained, and finally (2-3) authorization is made by comparing the organizations and positions of the requesting user with the UA expressions.

***(2) Authorization-on-authentication method***

A list of permitted bossiness applications is constructed at the time a user is authenticated. Upon the timing one of these applications is activated, the list is referenced and authorization decision is made. We need to measure the time for constructing the list of permitted business applications. This authorization method is suitable for log-on to a portal site listing personalized applications, like EIP (Enterprise Information Portal).

・*Direct Link Method:* In this method, links between belonging group lists, which are obtained as a result of policy evaluation, and roles are provisioned to the ID management database. The following is executed during user authentication: (3-1) All the groups the user belongs to are retrieved, (3-2) roles approved for any of the business applications are retrieved, and (3-3) a list containing applications approved to the user is obtained from the direct links between those groups and roles.

・*Rule Based Hierarchical Organization Method:* There is no direct link indicating UA in the ID management database. Instead logical expressions are stored to the database. The following is executed during user authentication: (4-1) The organizations and positions of the authenticated user are retrieved, (4-2) roles approved for any of the business applications are retrieved, (4-3) UA expressions containing one of the roles as terms are retrieved, and finally (4-4) a list of applications approved to the user is constructed by comparing the organizations and positions of the user with the UA expressions.

Table 2 shows measurement results where an LDAP directory is used as the ID management database. The results show that the processing time of the rule based RBAC is 10 percent larger than that of the direct-link RBAC, while throughput has the opposite tendency.

We substitute these values for the following:

```
check_authentication(ts1, check_on_demand_authorization, 9.691).
check_authentication(ts2, check_on_demand_authorization, 9.942).
check_authentication(ts1, check_authorization_on_authentication,
74.872).
check_authentication(ts2, check_authorization_on_authentication,
79.561).
```

According to [11], authorization of existing privileges to new users occurs at the rate of 1.30 per year, per employee. Let us assume that 1 second is necessary for provisioning for a new user by a typical identity management product [16][17]. By assuming that the time for `provision_U` and `provision_UA` are equal, we obtain the following values.

```
provision_U(ts1,add_user, 500).
provision_UA(ts1,add_user,_,500).
provision_U(ts2,add_user,500.
provision_UA(ts2,add_user,_,0).
```

Assumption: 5 objects out of 10 objects are accessed per day, 200 days per year, and 10 users access concurrently.

*On-demand authorization*

```
  Productivity Loss of ts1 =
      1,000 (msec) * 5,000 * 1.3 +          % A2111 of Fault Tree
      9.691 (msec)  * 5 * 5,000 * 200       % A213
      = 54,955 sec
  Productivity Loss of ts2 =
      500 (msec) * 5,000 * 1.3 +            % A2111 of Fault Tree
      9.942 (msec) * 5 * 5,000 * 200        % A213 of Fault Tree
      = 52,960 sec
```

*Authorization on authentication*

```
Productivity Loss of ts1 =
    1,000 (msec) * 5,000 * 1.3 +          % A2111 of Fault Tree
    74.872 (msec)  * 5,000 * 200          % A213 of Fault Tree
    = 81,372 sec
Productivity Loss of ts2 =
    500 (msec) * 5,000 * 1.3 +            % A2111 of Fault Tree
    79.561 (msec) * 5 * 5,000 * 200       % A213 of Fault Tree
    = 82,811 sec
```

The results are interpreted differently depending on the standpoint to tolerable risk. For example, the direct link method ts1 is justifiable if one-day delay can be accepted, or if creating accounts one day early and delaying release of them can be accepted. For the rule-based method ts2, it can be argued that 10 percent delay in processing time can be accepted and may not be regarded as productivity loss. This comes from the observation that while the total loss of productivity is obtained by the product of the amounts of employees and business days times 10 percent, the loss per user is a few seconds per day, which can be tolerated.

## 5 Conclusion

In this paper, we discussed extensions to RBAC to deal with circumstances of large enterprises, such as complexity of organizations, geographical diversities and heterogeneity of devices such as physical access control. To deploy RBAC for improving operability, convenience of employees, and security, we discussed designs of rule-based RBAC utilizing personnel affair information. Furthermore, to select most cost-effective RBAC extensions for enterprise-wide requirements, we proposed a quantitative risk evaluation method based on fault trees. We demonstrated usefulness of this RBAC risk evaluation method through application to the rule-based RBAC model, and presented cases where this model becomes advantageous. We think that this method is also applicable to enterprise-wide governance systems.

## References


1. Feraiolo,D. and Kuhn,R., Role-Based Access Control, Communications of the 15th NIST-NSA National Computer Security Conference, 1992.
2. Ferraiolo,D., Sandhu,R., Gavrila,S., and Kuhn,R., Proposed NIST standard for Role-Based Access Control, ACM Transaction on Information and System Security, Vol.4 No.3, 2001.
3. Feraiolo,D., Kuhn,R., and Chandramouli,R., Role-Based Access Control Second Edition, Computer Security Series, ARTECH HOUSE, 2007.
4. Kern,A., Kuhlmann,M., Schaad,A., and Moffett,J., Observations on the role life-cycle in the context of enterprise security management, SACMAT'02, 2002.
5. Kern,A., Kuhlmann,M., Kuropka,R., and Ruthert,A., A meta model for authorisations in application security systems and their integation into RBAC administration, SACMAT'04, 2004.
6. Al-Kahtani, M. A. and Sandhu, R., A Model for Attribute-Based User-Role Assignment, 18th Annual Computer Security Applications Conference (ACSAC), 2002.
7. Kern,A. and Walhorn,C., Rule support for role-based access control, SACMAT'05, 2005.
8. Zhang,L., Ahn,G., and Chu,B. A rule-based framework for role-based delegation and revocation ACM Transactions on Information and system security (TISSEC), 2003.

9. Byun,J., Soh,Y., and Bertino,E. Systematic Control and Management of Data Integrity, SACMAT'06, 2006.
10. Bank for International Settlements (BIS), Basel II: Revised international capital framework, 2004.
11. Gallaher,M., O'Connor,A, and Kropp,B. The Economic Impact of Role-Based Access Control (NIST Planning Report 02-1), March 2002.
12. Briney,A., Security Focused, Information security, September 2000.
13. Computer Security Institute, CSI Survey 2007, The 12th Annual Computer Crime and Security Survey, 2007.
14. U.S. Nuclear regulatory Commission, Fault Tree Handbook, January 1981.
15. Brooke, P., and Paige, R., Fault trees for security system design and analysis, Computer & Security, Vol.23, No 3, 2003.
16. Sun Java System Identity Manager.
http://www.sun.com/software/roducts/identity_mgr/
17. IBM Tivoli Identity Manager.
http://www.ibm.com/software/tivoli/products/identity-mgr/