情報処理学会論文誌 Vol. 50 No. 11 2727–2739 (Nov. 2009)

# 異種分散環境における ロールベースアクセス制御の定量的リスク評価

近藤誠一[†1,†2] 岩井原瑞穂[†3]
吉川正俊[†2] 虎渡昌史[†1]

企業活動において IT システムの重要インフラ化が進み，これまでの個別セキュリティ対策から企業統制への展開が進んでいる．企業統制の中で重要となるアクセス制御に関しては，ロールベースアクセス制御（RBAC）モデルに基づくシステムやセキュリティ製品が企業向けに実践されつつある．特に，地理的な分散や，物理セキュリティを含む異種システムの統合を課題とする大企業向けに，環境に応じたさまざまな拡張が行われている．本論文ではそれらの RBAC 拡張モデルを，定量的に評価するための手法について示す．大規模企業の要求に応じてコスト的に効果的な拡張モデルを選択するために，本論文では，故障木に基づく定量的リスク評価手法を提案する．トップ事象として，セキュリティ違反，機会損失，システム管理コストを，中間事象，基本事象としてセキュリティインシデント，RBAC 標準関数とそれらの詳細操作を持つ故障木を作成した．故障木の事象の発生確率は評価対象の RBAC 拡張モデルを適用した環境をもとに計算される．この手法を実際に企業で利用されている RBAC 拡張システムに適用した例を示し，セキュリティ向上，コスト低減の評価に効果的であることを示す．

# Quantitative Risk Evaluation of Role-Based Access Control for Heterogeneous Distributed Environment

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

Total and centralized security management of large enterprises has been in focus, due to the increasing demands on corporate governance. 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 securities. In this paper we propose quantitative risk evaluation method for their RBAC extension models. In order 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, productivity loss, and system management const as top events, and RBAC standard functions and security incidents as intermediate and basic events. Probabilities of top events are computed for given RBAC models and operation environments. We apply this method to two new RBAC extensions and confirm that these two extensions are more safer and cost effective than the base RBAC model.

## 1. はじめに

IT システムが企業活動の根幹となった現在，セキュリティリスク対策は企業存続のために必須項目となった．外部からの攻撃，内部からの情報漏洩に対して，これまで個別リスク対策がとられてきた．一方で，法制度，社会的責任の観点から，エンタープライズレベルの統制に基づく対策が整備されてきている．

セキュリティ統制のため，アイデンティティとアクセス制御の集中管理が運用面，統制面で注目されている．アクセス制御の手法としてロールベースアクセス制御（RBAC: Role-Based Access Control）モデル[1)–3)] によって設計されたアクセス制御ポリシが，企業情報システムで実践されつつある．しかし，広域分散拠点に散在する多様なセキュリティ対象へ展開されたアクセス制御情報の一貫性制御が課題となっており，システム管理コスト，従業員の生産性と脅威レベルのトレードオフを考慮した方式が必要とされている．

広域分散された全社システムでは，採用・退職，人事異動，設備の導入，セキュリティポリシの見直しが発生した際，影響する権限を計算して，その結果を関係するシステムに配布する必要がある．しかし，計算，配布には運用を含めた制約により，遅延が生じる．権限の付与遅延は，許可者の機会損失を招き，権限の抹消遅延は，未許可者のアクセス違反を招くリスクがある．このようなリスク対策として，RBAC モデルは，多様なターゲットシステムへの適用を想定したエンタープライズ RBAC[4),5)]，アクセスポリシをルールで表現する

†1 三菱電機インフォメーションシステムズ株式会社
Mitsubishi Electric Information Systems Corporation
†2 京都大学
Kyoto University
†3 早稲田大学
Waseda University

ルールベース RBAC[6)–8)] 等，本論文で提案する手法を含めて，さまざまなモデルが提唱されている．しかし，共通の評価手法がなく，定性的な比較，もしくは，特定システムでの性能評価[9)] にとどまっていたため，容認可能なコストでシステム要件に適合した RBAC モデルを特定することは困難であった．

本論文では，(1) セキュリティ事件/事故[10)–12)]，RBAC の関数群をもとにしたモデル共通の故障木[13)] の作成，(2) (1) で網羅的に作成された故障木の該当部分を選択し，AND-OR の関係を展開した論理型言語 Prolog のプログラムの作成，(3) ターゲットシステムごとの実測値，アンケート等で収集されたデータを (2) のプログラムのファクトに代入した解析による評価手法について示す．故障木を利用してセキュリティ解析を行う例[14)] はあるが，RBAC モデルを故障木で解析した先行事例はない．また，本論文では，文献 15) で述べた故障木を用いた基本概念にシステム管理コスト，許容時間を加え，2 つの新たな RBAC モデルを対象に提案手法を適用し，セキュリティ向上，コスト低減の評価に効果的であることが分かった．

2 章において，セキュリティポリシに則ったアクセス制御情報を設定するための関連技術について示す．3 章では，故障木を用いた RBAC とその拡張モデルの定量的リスク評価手法について示す．4 章では，企業における組織情報を利用したルールベースの RBAC モデル，および，ユーザが設定されていない IC カードを表現することを許す仮想ユーザを主体とする仮想ユーザ RBAC モデルを示し，3 章で提案した手法を適用した評価を行う．5 章で本論文をまとめる．

## 2. 関 連 技 術

本章では，RBAC モデルの基本定義，我々のリスク評価で用いる関連技術について示す．

### 2.1 RBAC モデルとその拡張

RBAC モデル[1),2)] では，ユーザ情報と，セキュリティ対象を，ロールを介して間接的に設定する．その効果として，ユーザ情報と，セキュリティ対象の変更管理を独立して行うことができる点があげられる．**図 1** に示した RBAC の標準である NIST（National Institute of Standards and Technology）のコア RBAC[3)] は以下のように定義されている．

定義 1．コア RBAC

- $USERS$，$ROLES$，$OPS\ and\ OBS$，
  ユーザ，ロール，操作，対象．
- $UA \subseteq USERS \times ROLES$，
  ユーザとロールの多対多のマッピング関係．

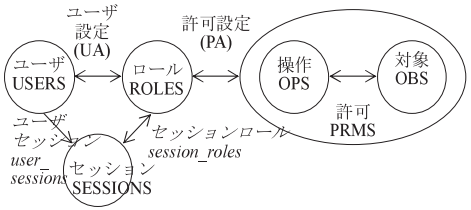


**図 1**　NIST コア RBAC
Fig. 1　NIST Core RBAC.

- $assigned\_users(r) = \{u \in USERS \mid (u, r) \in UA\}$，
  ロール $r$ のユーザ集合へのマッピング．
- $PRMS = 2^{(OPS \times OBS)}$，
  許可の集合．
- $PA \subseteq PRMS \times ROLES$，
  許可とロールの多対多のマッピング関係．
- $assigned\_permissions(r) = \{p \in PRMS \mid (p, r) \in PA\}$，
  ロール $r$ の許可の集合へのマッピング．
- $SESSIONS$，
  セッションの集合．
- $session\_user(s\colon SESSIONS) \rightarrow USERS$，
  セッション $s$ のユーザへのマッピング．
- $session\_roles(s_i) \subseteq \{r \in ROLES \mid (session\_user(s_i), r) \in UA\}$，
  セッション $s_i$ のロールの集合へのマッピング．
- $avail\_session\_perms(s\colon SESSIONS) \rightarrow 2^{PRMS}$，
  セッション $s$ 内でユーザに有効な許可．

コア RBAC では，これらの定義を維持管理するための関数である管理コマンドと，実行制御を行うシステム関数が定義されている．

(1) コア RBAC の管理コマンド

- $AddUser(user\colon NAME)$
- $DeleteUser(user\colon NAME)$
- $AddRole(role\colon NAME)$
- $DeleteRole(role\colon NAME)$

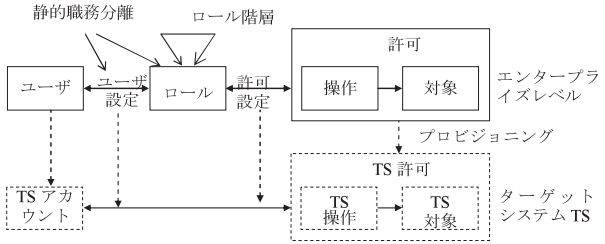


**図 2**　エンタープライズ RBAC モデル
Fig. 2　Enterprise RBAC Model.

- *AssignUser*(*user*, *role*: *NAME*)
- *DeassignUser*(*user*, *role*: *NAME*)
- *GrantPermission*(*object*, *operation*, *role*: *NAME*)
- *RevokePermission*(*operation*, *object*, *role*: *NAME*)

(2) コア RBAC のシステム関数

- *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*)

NIST では，さらに，ロールを継承させる階層を加えた階層 RBAC（Hierarchical RBAC）を定義している[3]．

企業の IT 環境におけるユーザとアクセス権の管理のために，エンタープライズ RBAC（ERBAC）モデルが提唱された[4),5)]．**図 2** に ERBAC モデルを示す．エンタープライズレベルで統合管理される情報を，複数のターゲットシステムに配布して企業全体のセキュリティポリシを維持管理する．このような維持管理のための配布をプロビジョニングと呼ぶこととする．エンタープライズレベルのユーザ設定 *UA* の定義には，通常，ルールが用いられる．文献 6) では，ユーザの持つ属性情報を要素とするルールを，実行時に解釈して認可するルールベース RBAC が紹介されている．ルールベース RBAC では，ユーザ属性の変更が生じても，*UA* を変更する必要がないため，運用コストが低減される．しかし，実行時の認可性能が新たな課題となる．文献 7) は，エンタープライズレベルでは，動的なルールベースで，ターゲットシステムでは，静的な設定とするモデルを示した．文献 8) では，ルールに基づく権利委譲とその抹消が，文献 9) では，アクセス制御に焦点を当てたデータの一貫性をメタレベルで表記してシステム的に管理する手法を示している．

### 2.2　セキュリティポリシのコストとリスクの定量化

企業においてセキュリティ全般統制が進展しない理由の 1 つとして，IT 事故発生のリスクが明確でなく，適正な情報セキュリティ投資の判断が困難であるとの指摘があり，リスクの定量化が求められている．金融機関では，情報システム障害等を含むオペレーショナルリスクの定量化を加えた新 BIS 規制（バーゼル II）が公表された[16)]．文献 10) では，RBAC の経済的インパクトとして，従業員あたりの操作に対する効果（Operating Benefits）$OB_{it}$ を以下のように定量的に定義している．$i$ は活動を，$t$ は年を表す．

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

$OB_{it}$ = 従業員あたりの操作に対する効果

$AC_{it}$ = 従業員あたりのシステム管理コスト低減

$PB_{it}$ = 従業員あたりの生産性効果

$SB_{it}$ = 従業員あたりのセキュリティ効果

さらに，RBAC のエンドユーザ効果として，以下の 3 点をあげている．

(1)　情報システム管理部門の運用処理時間の低減

(2)　従業員の生産性向上

(3)　セキュリティ違反の頻度と深刻度の低減

(3) のセキュリティ違反を具体的に調査した結果として，文献 11)，12) を参照して網羅性を強化している．(1) に関しては，以下の高頻度行動についてアンケートに基づいた RBAC と非 RBAC の比較を行っている．

- 新しいユーザへの既存の権限設定
- 既存ユーザの権限の変更
- 既存ユーザのための新しい権限の作成
- 権限の停止

文献 17) における被害量算定モデルでは，情報システムが停止した場合でも売上高等に影響が生じない許容時間を導入し，当該時間内であれば復旧によって被害は回復可能として総リスク量から除外する許容軽減リスク量を定めている．

### 2.3　故障木を利用したセキュリティシステムの設計・解析

故障木解析 FTA（Fault Tree Analysis）[13)] は，主に，原子力発電所，航空機，人工衛星といった安全性が重視されるシステムの設計，開発で用いられてきた．昨今，情報システム

の障害解析，セキュリティ解析へ適用する例が示されている[14),18),19)]．しかし，これらはセキュリティ全般の脅威分析として用いており，RBAC モデルを構成するデータ，操作まで詳細化した解析は行っていないため，従業員の人事異動，セキュリティポリシの変更といった運用が及ぼす影響を解析することができなかった．本論文では，文献 12) で評価を行っているネットワークアクセスの内部不正利用等，20 タイプの脅威を故障木の事象に取り入れるとともに，これらの中から RBAC が対象とする脅威として，不正アクセスに焦点を当てて，RBAC モデルに基づくアクセス制御システムへの適用について解析する．さらに，RBAC の管理コマンド，システム関数を詳細化し，セキュリティ違反，機会損失，システム管理コストをトップ事象として分析可能な故障木を作成し，それらを基盤として定量的評価を行う手法を示す．RBAC モデルを故障木で解析する場合，2.1 節で示した種々の拡張モデルを構成するデータの差異，ルールベース等の手続きの差異，大規模に代表される異種システム混合で大域分散されるシステムのデータの均一性を保つ運用管理を考慮することが課題となる．

## 3. 故障木を用いた RBAC モデル適用システムの定量的解析手法

### 3.1 解析の全体構成

これまで，種々の RBAC モデルが提唱されてきたが，それらの評価は個別要求仕様に対する有用性を示す手法がとられていた．本章では，**図 3** に示すように，さまざまな RBAC 拡張モデルを，シングルサインオンシステム，入退室管理システム等を含むエンタープライズレベルのセキュリティシステムに適用した場合の総合的な定量的解析手法を提案する．すなわち，さまざまな RBAC 拡張モデルを適用したシステムに共通な定量的評価基盤を定め，具体的なシステム固有の情報を代入して比較評価できる手法を考案した．本手法では故障木を採用することにより，3.3 節で示す RBAC 拡張モデルを適用したシステムを構成する要素に分解し，それらの構成要素を事象とすることにより，トレードオフの総合評価が可能となった．

### 3.2 解析の基本ステップ

セキュリティリスクには，具体的な損害を引き起こす直接的なものと，具体的な損害を防止するために組織で定められた間接的なポリシ違反に分類することができる．間接的なポリシに違反している状態で内外からの攻撃にあった場合，直接的な損害につながる．そこで，攻撃の確率を一定と仮定して，損害発生確率は，ポリシに違反している確率，すなわち，単位時間あたりの，違反状態の時間によって決まるものとする．

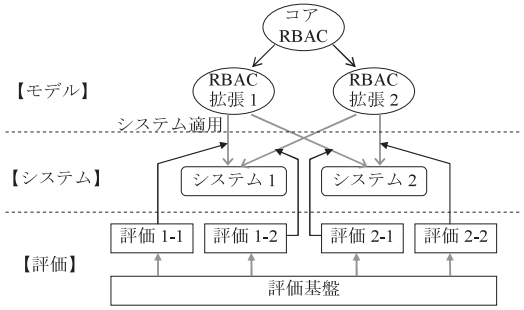


**図 3**　解析手法の全体構成
Fig. 3　Architecture of evaluation method.

本手法は以下に示す 4 ステップから構成される．

(1)　RBAC モデルに共通の故障木の作成：“セキュリティ違反”，“機会損失”，“システム管理コスト” をトップ事象とし，RBAC モデル，およびその拡張モデルの関数を中間事象，基本事象とする故障木を定義する．

(2)　モデル固有の故障木の作成：(1) で定義された共通故障木の中から与えられたモデルに該当する事象を選択する．

(3)　システム構成と構成要素の性能パラメータの定義：評価対象システムに応じて基本事象，中間事象の発生確率を数値化する．本論文のモデルでは，発生確率は，たとえば 1 日といった単位時間に占める応答時間の割合に比例するものとする．

(4)　リスク評価関数の実装：(2)，(3) の結果を論理型言語 Prolog のプログラムで定義して計算可能とする．

### 3.3 対象とするアクセス制御システム

集中管理されたアクセス制御ポリシに基づいて企業全体の統制を行うシステム構成が，一般的となっている．本論文では，異種分散環境にある企業全体の統合アクセス制御システムに必要とされる RBAC の関数の機能について考察する．以下の構成要素からなっているものとする．

- エンタープライズレベルシステム：エンタープライズレベルのセキュリティポリシに則り，ユーザ情報，アクセス制御情報を管理する．
- ターゲットシステム：エンタープライズレベルのユーザ情報，アクセス制御情報を登録して，業務に応じたアクセス制御を実行する．

- 認証・認可情報提供システム：各ターゲットシステムで必要とするユーザ情報，アクセス制御情報を，個々のターゲットシステムに適した方式で提供する．提供方式は，事前一括提供，変更時即時提供のいずれかの手法がとられるものとする．

次に，エンタープライズレベルのデータベースに対する変更情報をターゲットシステムへ反映する際の時間コストの評価要素を示す．システムは以下の情報を管理すると仮定する．

(1)　ユーザ情報 *USERS*
　(1-1)　ユーザ識別子
　(1-2)　ユーザ属性（パスワード，保有 IC カード，生体情報，証明書等）
　(1-3)　ユーザグループ（ユーザに付随して与えられる構造，たとえば，組織，プロジェクト等）
(2)　エンタープライズレベルロール *ROLES*
(3)　エンタープライズレベルユーザ設定 *UA*
(4)　エンタープライズレベル許可 *PRMS*
(5)　エンタープライズレベル許可設定 *PA*
(6)　ターゲットシステムレベル個別ロール *TS ROLES*
(7)　ターゲットシステムレベル個別許可 *TS PRMS*
(8)　ターゲットシステムレベル個別許可設定 *TS PA*

システム管理コマンドが起動されると，エンタープライズレベルに対する変更情報をターゲットシステムにプロビジョニングして，情報の同期をとる．モデル固有の同期方式を評価するために，プロビジョニングすべき要素の計算と (1)〜(5) の情報ごとのプロビジョニングにタスクを細分化して分析する．(1)〜(5) のすべてをターゲットシステム ts1 にプロビジョニングする場合を**図 4** に，エンタープライズレベルのロールを解釈して (3)，(4)，(5) をまとめてターゲットシステム ts2 に反映する場合を**図 5** に示す．図 5 では，エンタープライズレベルで更新が生じると，(1)〜(5) を解釈して，以下に示すユーザと許可の関係 *UP* を求める．

- $UP \subseteq USERS \times PRMS$,
  ユーザと許可の多対多のマッピング関係.
- $user\_permissions(u) = \{p \in PRMS \,|\, (u,p) \in UP\} = \{p \in PRMS \,|\, (u,r) \in UA \wedge (p,r) \in PA\}$.

このように網羅的に整理したプロビジョニングのタスクから，評価対象のモデルの操作に関連したタスクを選択し，各性能コストを指定する．たとえば，ユーザ追加操作は以下のタスクを選択する．

① ルールに該当するユーザ設定 *UA* を計算
② 追加されたユーザ情報をプロビジョニング
③ ① の結果であるユーザ設定をプロビジョニング

以下に示す論理プログラムを用いて，各タスクの性能コストを指定する．

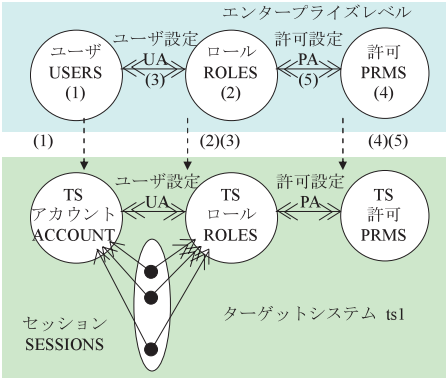


**図 4**　プロビジョニング情報（1）
Fig. 4　Provisioning information (1).

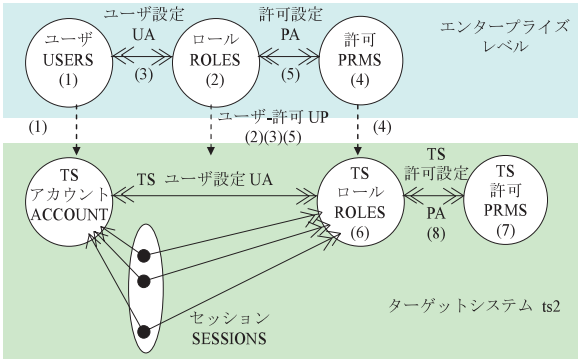


**図 5**　プロビジョニング情報（2）
Fig. 5　Provisioning information (2).

```
:- evaluate_provisioning(U_ID, U_Attr, APP_ID, add_user, Max_time,
Total_time).

evaluate_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, UP_List, PA_calculation_time),
    provision_U(APPID,FuncName,U_time),     % (1)
    provision_R(APPID,FuncName,R_time),     % (2)
    provision_P(APPID,FuncName,P_time),     % (4)
    provision_UA(APPID,FuncName,UA_List,UA_time), % (3)
    provision_PA(APPID,FuncName,PA_List,PA_time), % (5)
    provision_UP(APPID,FuncName,UP_List,UP_time), % (2)(3)(5)
    max([U_time,R_time,P_time,UA_time,PA_time,UP_time], Max_time),
    sum([U_time,R_time,P_time,UA_time,PA_time,UP_time], Total_time).

% provision_U(in,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).
provision_UP(ts1,add_user,_,0).

provision_U(ts2,add_user, (measured_users_provisioning_time)).
provision_R(ts2,add_user, 0).
provision_P(ts2,add_user,0).
provision_UA(ts2,add_user,_,0).
provision_PA(ts2,add_user,_,0).
provision_UP(ts2,add_user,UP_List,(measured_up_provisioning_time)).
```

### 3.4　共通 RBAC 故障木を用いたリスク解析

故障木を用いて RBAC モデル共通のセキュリティリスク解析を行う．はじめに，RBAC モデルに基づくアクセス制御システムにとって低減させるべきトップ事象として，3.1 節で示した以下の 3 点をあげる．

TE1 セキュリティ違反：権利が与えられていないユーザが操作できる確率．すなわち，単位時間内で，危険にさらされている時間の割合を示す．

TE2 機会損失：権利が与えられているユーザが操作できない確率．すなわち，単位時間内で，従業員が操作できない時間の割合を示す．

TE3 システム管理コスト：異種分散環境において変更情報の同期を行っている確率．すなわち，単位時間内で，計算，プロビジョニング等でシステムが稼動している時間の割合を示す．

TE1，TE2 は TE3 と密接な関係にある．TE3 の確率を下げること，および TE3 の影響を受けないようにすることが，TE1，TE2 の確率を下げることに通じる．

上記トップ事象に対して，その最終要因となる基本事象 $X_i$ $(i = 1, \cdots, n)$ を含む AND-OR 木を構成する．RBAC 拡張モデルに基づいて作成した大規模企業向けセキュリティシステムの故障木の概要を**図 6** に示す．上部の全体像で構成要素を示し，下部でその中から RBAC 拡張モデルに関連する事象を拡大して示した．中間事象として，(1) 文献 10)–12) を参照して網羅的に配置したセキュリティリスク，(2) 文献 2) の管理コマンド，システム関数の処理時間内に生じるリスクを採用した．*UA*，*PA* に関するルールを評価する手続き，およびユーザ情報，ロール，許可といった基本情報のプロビジョニングを中間事象，基本事象として選択し，各モデルによって異なるタスクまで詳細化してその処理時間の差異を解析できるようにした．

TE1 セキュリティ違反の上位レベルの事象は網羅性を考慮して一般に要求されるセキュリティリスクを機械的に配置した．CSI Computer and Security Survey [12] で示されたリスクを導入し，RBAC に直接影響を与えるリスク中間事象 A111 と A112 に焦点を当てた．これらの事象が単位時間あたりに発生する確率は一定であると仮定すると，RBAC 関数実行中に発生する不正な権限が存在する時間を短縮することによってセキュリティを向上させることが可能となる．TE3 システム管理コストの上位レベルの事象は NIST RBAC の経済インパクト [10] で定量的評価が行われた項目で，NIST RBAC の管理コマンドを網羅している．

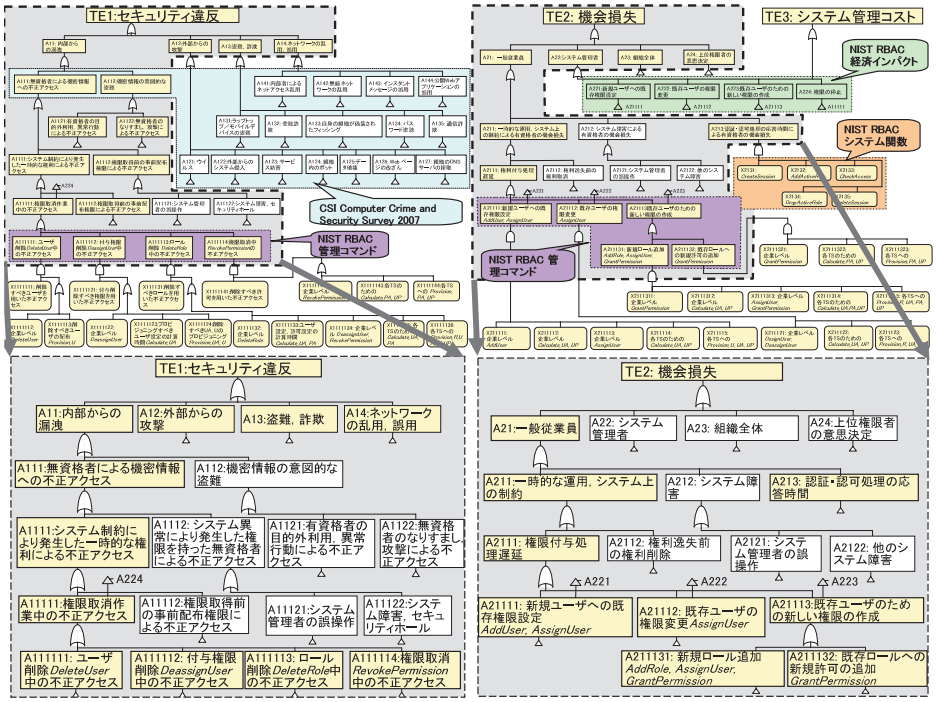


**図 6**　共通 RBAC 故障木
Fig. 6　Common RBAC Fault Trees.

## 4. RBAC 拡張モデルの適用とその定量的評価

本章では，最初に，大規模分散された企業規模のセキュリティシステムのアクセス権限を統制するために最適化した以下に示す新たな 2 つの RBAC 拡張モデルを提案する．次に 3 章で示した評価手法を用いて，それらのモデルをシステム適用した場合の評価を行う．最後に，評価手法の有効性を示す．

(1)　Web アプリケーションのシングルサインオンにおいて，プロビジョニング一括実効コストを低減させるルールベース階層組織 RBAC モデル

(2)　ユーザ設定を行わないで登録された IC カードを用いて変更時における実時間プロビジョニングコストを低減させる仮想ユーザ RBAC モデル

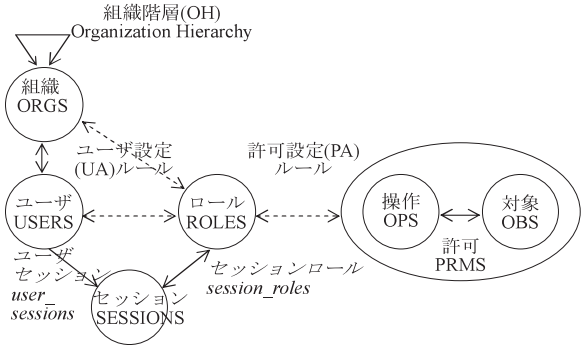


**図 7**　ルールベース階層組織 RBAC
Fig. 7　Rule-based hierarchical organization RBAC.

これらの実システムへの適用モデルに，3 章で示した定量的評価手法をケーススタディとして適用する．

### 4.1 ルールベース階層組織 RBAC への適用

#### 4.1.1 モデル定義

ターゲットシステムとして，コア RBAC とルールベース階層組織 RBAC を比較して解析を行う．Web 業務アプリケーションでは，ユーザ情報，*UA*，*UP* といった，アクセス制御情報を更新するためのプロビジョニングを必要とする．これに対応するため，**図 7** に示す組織をロールから独立させて，実行時にルールを解釈するルールベース階層組織 RBAC モデルを提案する．この拡張モデルに基づくターゲットシステムを ts3 とする．ts3 では，プロビジョニング時には，ポリシを示すルールを評価せず，ルール自身をプロビジョニングし，実行時にルールを解釈して，認証・認可を決定する．したがってユーザ追加時のプロビジョニング時間は，以下に示すように *UA* は 0 となる．

```
provision_U(ts3,add_user, (measured_users_provisioning_time)).
provision_UA(ts3,add_user,_,0).        % ts1 と異なる部分
```

2 種類のモデルの定義を以下に示す．

(a) コア RBAC に基づく直接リンク方式

［システム条件］事前にルールを評価してユーザとロールの直接の組合せをプロビジョニング

［評価する操作］

設定時タスク：`calculate_UA`，`provision_U`，`provision_UA`

認証時タスク：Check_Access

(b) ルールベース階層組織方式

［システム条件］ユーザ情報のみプロビジョニングし，実行時に *UA* を計算して判断

［評価する操作］

設定時タスク：provision_U

認証時タスク：calculate_UA, Check_Access

［ts1 と ts3 のモデル定義］故障木の A1111，A2111，A213

```
:- 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, ts3, add_user,
Time_of_Add_user_to_ts3).  % A1111 and A2111 for ts3
:- check_authentication(ts1, check_on_demand_authorization,
    Time_of_check_authentication_1).          % A213 for ts1
:- check_authentication(ts3, check_on_demand_authorization,
    Time_of_check_authentication_2).          % A213 for ts3
:- check_authentication(ts1, check_authorization_on_authentication,
    Time_of_check_authentication_1).          % A213 for ts1
:- check_authentication(ts3, check_authorization_on_authentication,
    Time_of_check_authentication_2).          % A213 for ts3
```

#### 4.1.2 評　　価

本項では，4.1.1 項のモデルをシングルサインオンシステムに適用した場合のモデル間の比較評価を 3 章で示した評価手法を用いて行うこととする．3.2 節で示した定義に実測値を適用してその妥当性を考察する．ユーザ数 5,000 名，組織の階層を 5 階層とする．**表 1** にデータ構造を示す．ロールは，すべての組織，役職ごとに自動生成し，そのうち，16 個を認可可能として対象に設定するものとする．ユーザ設定のルールを示す条件式は，これまでの実装例から，1〜2 項目程度の論理積または論理和からなるものと仮定する．認可判定として，以下の 2 種類について，応答時間，スループットを 2 台のサーバを用いて測定した．

- 認証認可サーバ：Pentium4 2.66 GHz, メモリ 1 GB
  Windows 2003 Server
- ディレクトリサーバ：Pentium4 3.0 GHz, メモリ 1 GB
  SunOne Directory Serverf 5.2, Windows 2003 Server

**表 1**　測定用データ構造
Table 1　Data structure for measurement.

(a) 組織構造とユーザ数

| | 組織数 | ユーザ数 | 役職 |
|---|---|---|---|
| レベル 1 | 4 | 4 | 本部長 |
| レベル 2 | 6 | 48 | 事業所長 |
| レベル 3 | 150 | 450 | 部長 |
| レベル 4 | 450 | 4,500 | 課長、担当 |
| 計 | 610 | 5,002 | |

(b) 認可設定, ユーザ数設定

| | |
|---|---|
| 対象数 | 10 |
| 1 対象に設定するロール数 | 2 |
| 対象に設定されたロール総数 | 16 |
| UA ルール数 | 2 |
| 所属グループ数 | 10 |

(1) 実行時認可判定

Web アプリケーションごとに初回起動時に認証ユーザが権限を持つか否かの認可判定を行う．

- 直接リンク方式

ユーザが所属するグループ，グループとロールのリンクをターゲットシステムにプロビジョニングする．Web アプリケーションの起動時に，(1-1) 起動が許可されているロールを求め，(1-2) 所属グループリストを直接比較して認可判定を行う．

- ルールベース階層組織方式

ターゲットシステムには，*UA* ルールを示す論理式を格納する．ユーザ認証時に，上位を含めた所属組織，役職等の属性情報を参照して保持する．Web アプリケーション起動時に，(2-1) 起動が許可されているロールを求め，(2-2) そのロールが保持する *UA* ルールを求め，(2-3) ユーザの属性情報と *UA* ルールの比較により認可判定を行う．

(2) ユーザ認証時認可判定

ユーザ認証時に起動可能な Web アプリケーションのリストを作成し，起動時に参照して認可判定を行う．このリスト作成時間を性能評価対象とする．

- 直接リンク方式

ユーザが所属するグループ，グループとロールのリンクをターゲットシステムにプロビジョニングする．ユーザ認証時に，(3-1) 所属するすべてのグループを参照し，(3-2) 各

**表 2**　性能測定結果
Table 2　Measurement results.

| | 実行時認可判定 | | ユーザ認証時認可判定 | |
|---|---|---|---|---|
| | 応答時間 (msec) | スループット (transactions /sec) | 応答時間 (msec) | スループット (transactions /sec) |
| 直接リンク | | | | |
| 単独 | 1.509 | 662.856 | 11.363 | 90.277 |
| 10 多重 | 9.691 | 1,031.914 | 74.872 | 133.561 |
| ルールベース | | | | |
| 単独 | 1.575 | 645.856 | 12.210 | 81.486 |
| 10 多重 | 9.942 | 1,005.874 | 79.561 | 125.690 |

Web アプリケーションについて許可されているロールを参照し，(3-3) グループとロールの直接リンクから認証されたユーザに許可されている Web アプリケーションのリストを求める．

- ルールベース階層組織方式

ターゲットシステムには，*UA* ルールを示す論理式を格納する．ユーザ認証時に，(4-1) 認証されたユーザの所属組織，役職等の属性情報を求め，(4-2) 各 Web アプリケーションについて許可されているロールを参照し，(4-3) 各ロールの属性として持つ *UA* ルールを求め，(4-4) ユーザの属性情報と *UA* ルールの比較により，認証されたユーザに許可されている Web アプリケーションのリストを求める．

測定結果を**表 2** に示す．ルールベース階層組織方式の応答時間，スループットは，直接リンク方式と比較して，最大で 10%程度であることが分かった．これらの値を定量値として以下のように代入する．

```
check_authentication(ts1, check_on_demand_authorization, 9.691).
check_authentication(ts3, check_on_demand_authorization, 9.942).
check_authentication(ts1, check_authorization_on_authentication, 74.872).
check_authentication(ts3, check_authorization_on_authentication, 79.561).
```

文献 10) の統計データによると年間 1 従業員あたり，新規ユーザへの既存権限の付加は 1.30 件である．典型的なアイデンティティ管理製品の性能データをもとに，新規ユーザのプロビジョニング性能を 1 秒/人と仮定する．`provision_U` と `provision_UA` の性能は等しいと仮定して，以下のように定量値を代入する．

```
provision_U(ts1, add_user, 500).
provision_UA(ts1, add_user,_, 500).
provision_U(ts3, add_user, 500).
provision_UA(ts3, add_user,_, 0).
```

1 日に 10 対象のうち，5 対象にアクセス，200 日/年，10 ユーザ同時アクセスと仮定して解析する．

(1) 実行時認可判定

```
ts1 の TE2:機会損失 =
    1,000 (msec) * 5,000 * 1.3 +      % 故障木の A2111
    9.691 (msec) * 5 * 5,000 * 200    % 故障木の A213
    = 54,955 sec
ts3 の TE2:機会損失 =
    500 (msec) * 5,000 * 1.3 +        % 故障木の A2111
    9.942 (msec) * 5 * 5,000 * 200    % 故障木の A213
    = 52,960 sec
ts1 の TE3:システム管理コスト =
    1,000 (msec) * 5,000 * 1.3 +      % 故障木の A2111
    = 6,500 sec
ts3 の TE3:システム管理コスト =
    500 (msec) * 5,000 * 1.3 +        % 故障木の A2111
    = 3,250 sec
```

(2) ユーザ認証時認可判定

```
ts1 の TE2:機会損失 =
    1,000 (msec) * 5,000 * 1.3 +      % 故障木の A2111
    74.872 (msec) * 5,000 * 200       % 故障木の A213
    = 81,372 sec
ts3 の TE2:機会損失 =
    500 (msec) * 5,000 * 1.3 +        % 故障木の A2111
    79.561 (msec) * 5 * 5,000 * 200   % 故障木の A213
    = 82,811 sec
ts1 の TE3:システム管理コスト =
    1,000 (msec) * 5,000 * 1.3 +      % 故障木の A2111
    = 6,500 sec
ts3 の TE3:システム管理コスト =
```

```
500 (msec) * 5,000 * 1.3 +        % 故障木の A2111
= 3,250 sec
```

以上の結果から，TE2：機会損失では，実行時認可判定ではルールベース階層組織方式が，ユーザ認証時認可判定では直接リンク方式が有利である，すなわち，被害発生確率は，違反状態の時間に対応していることから，リスクが小さいことが分かった．TE3：システム管理では，ルールベース階層組織方式が有利であるので，両者の優先度によって最終判断がなされる．

さらに，許容軽減リスク量を取り入れた評価を行う．直接リンク方式において，たとえば，翌朝反映までを許容時間と設定するとプロビジョニングによる機会損失は許容時間内に収まり，総リスク量から除外可能となる．また，ルールベース階層組織方式における応答時間は，認証時の 1 sec を許容時間と設定すると，許容時間内に収まり，総リスク量から除外可能となる．

### 4.2　仮想ユーザ RBAC への適用

#### 4.2.1　モデル定義

セキュリティ，利便性，コスト面で IC カードの普及が進展している．しかし，IC カードは，恒久的に利用可能なものではなく，紛失，破損等が原因で利用不可となったときに，速やかに代替手段に移行する必要がある．この作業が遅延した場合は，従業員の機会損失と，紛失カードによる不正アクセスのリスクが生じる．

IC カードを利用した代表的なセキュリティシステムとして，入退室管理システムがあげられる．入退室管理システムでは，3.3 節で示したターゲットシステム ts2 のようにロールを解釈した結果，すなわち，入退室管理システムの持つ，フロア，部屋等のレイアウトによってグループ化された入室可能な扉の集合へのマッピングがプロビジョニング対象となる．その結果，ユーザの追加，削除，変更が生じた場合，プロビジョニングの遅延により，不正アクセス，機会損失のリスクが発生する．そこで，通常は，非携帯者向け，来訪者向けにあらかじめ入室可能な予備カードを準備して，手交する方式がとられる．しかし，この行為は，従来の RBAC モデルでは十分に表現することはできない．**図 8** に IC カードを仮想ユーザとして，ユーザとは別に定義し，ターゲットシステムへは，この仮想ユーザをプロビジョニングする方式を提案する．この結果，非携帯者，来訪者に対して，プロビジョニングのコストをかけないで，セッションを開始することが可能となる．

ERBAC モデルと仮想ユーザ RBAC モデルを以下の仮定のもと比較する．

［システム条件］プロビジョニングは 1 日 1 回といったように定められた時間間隔で行う．

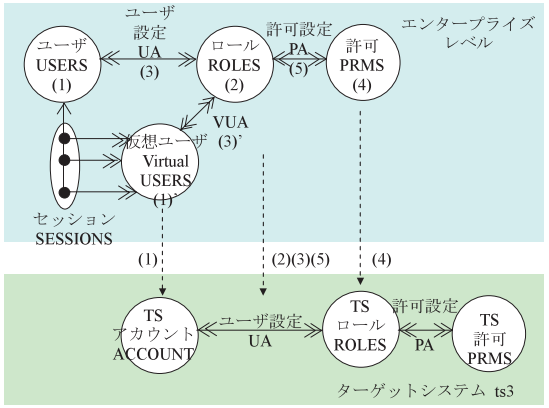


**図 8**　仮想ユーザ RBAC
Fig. 8　Virtual user RBAC.

［評価する操作］`change_user`

IC カード紛失申告時に，旧カードを停止し，新カードを利用可能にする．内部的には，旧カード情報の削除と新カード情報の設定を行う．

［ERBAC のモデル定義］故障木の A11112，A111112，A211112，A2112 が該当

```
:- evaluate_user_provisioning(user1, card_id(new_card_id), ts2,
      change_user_attributes, ERBAC_time).
        % カード ID 属性を新しいカードの ID に置き換え
If ERBAC_time > 0 then
      A111112: 旧カードによる不正アクセス
      A211112: 新カードによる機会損失
  else  A2112: 旧カードによる機会損失
      A11112: 新カードによる不正アクセス
```

［仮想ユーザ RBAC のモデル定義］：故障木の A111112，A11112，A21111，A2112 が該当

```
:- evaluate_user_provisioning(user1, card_id(new_card_id),
                    ts2, add_virtual_user, VURBAC_time_1),
  evaluate_user_provisioning(user1, [], ts2, del_virtual_user,
       VURBAC_time_2).
```

```
If VURBAC_time_1 > 0 then
        A21111: 新カードによる機会損失
  else  A11112: 新カードによる不正アクセス
If VURBAC_time_2 > 0 then
        A111112: 旧カードによる不正アクセス
  else  A2112: 旧カードによる機会損失
```

### 4.2.2 評　　価

本項では，4.2.1 項のモデルを入退室管理システムに適用した場合のモデル間の比較評価を 3 章で示した評価手法を用いて行うこととする．ERBAC では，紛失を認識した後，手続きを行うので，変更ユーザに対するプロビジョニングに要する平均時間は，ポーリング時間/2 である．ポーリング間隔を 24 時間とすると，平均時間は，12 時間となる．

仮想ユーザ RBAC では，あらかじめカードを準備するので，設定時間と実際に用いられる時間の差異は，負数で表現する．始業前，たとえば，午前 0 時に利用可能な予備カードの有効化を行うとすると，実際にユーザに設定するまでの平均時間は，−12 時間となる．仮想ユーザ RBAC の旧カードに対するプロビジョニング時間は，ERBAC と同様で，12 時間となる．

これらに従って，定量値を入力する．

```
provision_U(ts2,change_user_attributes, 12).
provision_R(ts2,change_user_attributes, 0).
provision_P(ts2,change_user_attributes,0).
provision_UA(ts2,change_user_attributes,_,0).
provision_PA(ts2,change_user_attributes,_,0).
provision_UP(ts2,change_user_attributes,_,12).
provision_U(ts2,add_virtual_user, -12).
provision_R(ts2,add_virtual_user, 0).
provision_P(ts2,add_virtual_user,0).
provision_UA(ts2,add_virtual_user,_,0).
provision_PA(ts2,add_virtual_user,_,0).
provision_UP(ts2,add_virtual_user,_,-12).
provision_U(ts2,del_virtual_user, 12).
provision_R(ts2,del_virtual_user, 0).
```

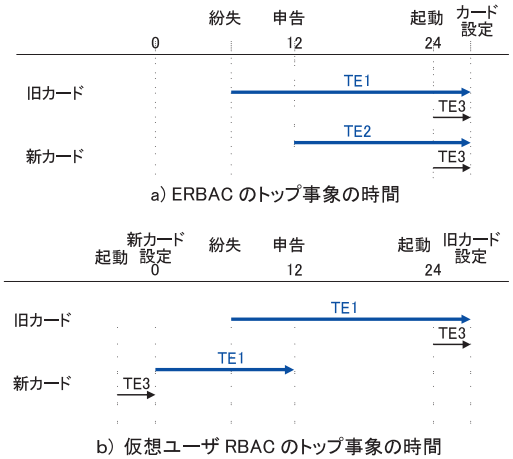


**図 9**　仮想ユーザ RBAC のトップ事象
Fig. 9　Top events of virtual user RBAC.

```
provision_P(ts2,del_virtual_user,0).
provision_UA(ts2,del_virtual_user,_,0).
provision_PA(ts2,del_virtual_user,_,0).
provision_UP(ts2,del_virtual_user,_,12).
```

これらの値をもとに故障木のトップ事象を評価した結果を以下に示す．システムとして関与できない紛失から申告までの時間は解析対象とはしないこととする．

- **図 9** (a) に ERBAC に関する故障木の各トップ事象の時間を図示する．TE1 セキュリティ違反には，旧カードについては紛失から申告までの時間＋12 時間＋設定システム時間が生じる．TE2 機会損失には，新カードについて，申告から設定まで 12 時間＋設定時間が生じる．TE3 システム管理コストには，新旧カード設定時間が生じる．
- 図 9 (b) に仮想ユーザ RBAC に関する故障木の各トップ事象の時間を図示する．TE1 セキュリティ違反の旧カードについては，紛失から申告までの時間＋12 時間＋設定システム時間が生じる．新カードについてはあらかじめ用意するカードを利用した不正アクセスのリスクとして，仮想ユーザへの設定から実ユーザへの貸出までの 12 時間が生じる．TE3 システム管理コストには，旧カードの設定時間，新カードの準備時間が生じる．

以上のことから，両者を比較すると，12 時間の新カードに関する TE2：機会損失と，12 時間の予備カードの TE1：セキュリティ違反，予備カード準備に要する TE3：システム管理のトレードオフが判断の材料となることが分かる．事前発行された予備カードが，厳重に人的管理されることが保証され，かつ事前カードの準備が自動化されれば，仮想ユーザが有利なことが分かる．

### 4.3 議　　論

このように RBAC 拡張モデルを採用した大規模企業システムに本論文で示したリスク定量的評価を適用した結果，以下の効果を得ることが可能となった．

- RBAC 標準の管理コマンド，システム関数，およびセキュリティ一覧[12] といった網羅的な項目を評価対象とすることにより，一括実行，変更時即時実行，実行時認証・認可まで，RBAC の経済的効果として示されている 3 種のトップ事象であるセキュリティ違反，機会損失，管理コストのそれぞれの項目について，RBAC 拡張モデル間の定量的な比較を行い，長所短所やトレードオフを明確化することができた．
- RBAC モデルで定義された関数を個別データのプロビジョニングレベルまで詳細化して，時間軸で定量的に解析することにより，異種分散環境にある大規模セキュリティシステムにおけるルールベースモデル，IC カードの運用といった全社規模のシステムで求められる機能の効果を，同一次元で比較評価することができた．

## 5. おわりに

本論文では，大規模企業に見られる複雑な組織構造，広域分散システム，入退室管理システム等の異種システムといった環境でのアクセス制御統合への RBAC 拡張モデルの適用とその定量的評価について示した．RBAC モデルの適用には，セキュリティ，従業員の操作性，システム構築・運用コストを考慮する必要がある．それらの判断を行うための定量的リスク評価方式として，セキュリティ違反，機会損失，システム管理コストをトップ事象として作成した故障木をもとにした定量的リスク評価方式を提案した．また，企業向け実システムへ適用されているルールベース階層組織 RBAC モデル，仮想ユーザ RBAC モデルを新たに提案し，それらを題材に，提案した方式を用いた解析を行った．実システム性能測定データをもとに，(a) RBAC 拡張モデル，およびプラットフォームの選択，(b) 提案モデルの優位性の提示，(c) 許容時間を考慮した解析に効果があることを示した．今後は，実システム性能測定が不要な動的シミュレーションを加えた解析に展開する所存である．

**謝辞**　本研究の一部は，科研費（21300034）および JST 戦略的国際科学技術協力推進事業の助成を受けている．

## 参 考 文 献

1) Ferraiolo, 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 Trans. Information and System Security*, Vol.4 No.3 (2001).
3) Ferraiolo, 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 authorizations in application security systems and their integration 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 Trans. 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) Gallaher, M., O'Connor, A. and Kropp, B.: The Economic Impact of Role-Based Access Control (NIST Planning Report 02-1) (2002).
11) Briney, A.: Security Focused, *Information Security* (2000).
12) Computer Security Institute: *CSI Survey 2007, The 12th Annual Computer Crime and Security Survey* (2007).
13) Vesely, W.E., Goldberg, F.F., Roberts, N.H. and Haasl, D.F.: Fault Tree Handbook, U.S. Nuclear Regulatory Commission (1981).
14) Brooke, P. and Paige, R.: Fault trees for security system design and analysis, *Computer & Security*, Vol.23, No.3 (2003).
15) Kondo, S., Iwaihara, M., Yoshikawa, M., et al.: Extending RBAC for Large Enterprises and Its Quantitative Risk Evaluation, *Towards Sustainable Society on Ubiquitous Networks*, *IFIP I3E2008*, *International Federation for Information Processing*, Vol.286, pp.99–112, Springer (2008).
16) Bank for International Settlements (BIS), Basel II: Revised international capital framework (2004).

17) 経済産業省：リスク定量化に関する検討資料，企業における情報セキュリティガバナンスのあり方に関する研究会報告書参考資料 (2003).
18) Sasaki, R., Ishii, S., Hidaka, Y., et al.: Development Concept for and Trial Application of a “Multiplex Risk Communicator”, *IFIP I3E2005*, pp.607–621, Springer (2005).
19) 佐々木良一，日高　悠，守谷隆史ほか：多重リスクコミュニケータの開発と適用，情報処理学会論文誌，Vol.49, No.9, pp.3180–3190 (2008).

(平成 21 年 1 月 30 日受付)
(平成 21 年 9 月 11 日採録)

**近藤　誠一**（正会員）

1984 年京都大学大学院工学研究科情報工学専攻修士課程修了．同年三菱電機（株）入社．1989〜1992 年（財）新世代コンピュータ技術開発機構（ICOT）出向．2009 年より三菱電機インフォメーションシステムズ（株）出向．情報セキュリティシステムに関する研究開発に従事．

**岩井原瑞穂**（正会員）

1988 年九州大学工学部情報工学科卒業．1990 年同大学院修士課程修了．1993 年同大学院博士課程修了．博士（工学）．同年より九州大学大学院総合理工学研究科助手．1995 年九州大学大学院システム情報科学研究科助教授．2001 年京都大学大学院情報学研究科助教授．2009 年早稲田大学大学院情報生産システム研究科教授．電子情報通信学会，ACM，IEEE，日本データベース学会各会員．

**吉川　正俊**（正会員）

京都大学大学院工学研究科博士後期課程修了．工学博士．京都産業大学，奈良先端科学技術大学院大学，名古屋大学を経て，2006 年より京都大学大学院情報学研究科教授．この間，南カリフォルニア大学客員研究員，ウォータルー大学客員准教授．XML データベース，異種情報源の統合等の研究に従事．電子情報通信学会，ACM，IEEE Computer Society 各会員．

**虎渡　昌史**（正会員）

1983 年慶應義塾大学大学院工学研究科電気工学専攻修士課程修了．同年三菱電機（株）入社．2006 年三菱電機インフォメーションシステムズ（株）に転籍．情報セキュリティシステムに関する研究開発に従事．