

異種分散環境におけるロールベースアクセス制御のモデル駆動設計手法

近 藤 誠 一^{†1} 岩井原 瑞穂^{†2} 吉 川 正 俊^{†3}
小 宮 崇^{†4} 虎 渡 昌 史^{†1}

官庁、企業等の組織体における機密情報漏えい防止、コンプライアンス対応として、人、機密情報、システム等、組織体を構成する要素、およびそれらの関係で定義されるアクセス制御管理が重要視されている。企業で定められたアクセス制御ポリシーを正しく実践するために、ロールベースアクセス制御（RBAC）モデルを用いて設計されたアイデンティティ・アクセス管理（IAM: Identity and Access Management）システムが企業向けに構築されている。IAM システムでは、人事異動等によってポリシーに反する状態が生じないように変更処理が行われるが、大企業に代表される広域分散環境では、プラットフォーム、運用管理等の制約によって一時的に生じるポリシーに反する状態を考慮する必要がある。しかし、異種分散システムを構成する各システムの RBAC モデルの差異、データ同期の方式の差異を統一的に扱うモデルが定義されていなかったため、過渡状態によってリスクが生じる時間等の定量的な全体評価が困難であった。本論文では、これまで提唱されてきた主要な RBAC モデルを組み合わせた異種分散 RBAC モデルを形式的に定義するために、管理者オペレーションを起点とし、RBAC の要素単位でデータ項目、操作を形式的に定義する手法を示す。さらに、システム設計結果にリスクが生じる時間情報を直接付加した性能評価モデルを作成し、設計段階におけるリスク評価、および評価結果をフィードバックする設計手法について示す。

Role-Based Access Control Model-Driven Design Method for Heterogeneous Distributed Environment

SEIICHI KONDO,^{†1} MIZUHO IWAHARA,^{†2}
MASATOSHI YOSHIKAWA,^{†3} TAKASHI KOMIYA^{†4}
and MASASHI TORATO^{†1}

Identity management of elements that construct the organization, such as employees, confidential information, IT systems, and access control defined by

relations of identities has been in focus for confidential information leakage prevention and compliance of governances and enterprises. In order to implement enterprise security policies, IAM (Identity and Access Management) systems based on the RBAC (Role Based Access Control) model have been widely introduced to enterprises. In IAM systems processing is performed to prevent the state against the policy. In widely distributed environments such as large enterprises, we must consider the temporal states against the policy by constraints of the platform and system management. In such a case comprehensive quantitative risk evaluation of transitional periods was difficult because standard definition methods for various RBAC models and data synchronization have not been established. In this paper we propose a standard RBAC definition method that can be applied to heterogeneous distributed systems constructed by major RBAC models. Also, we propose a risk evaluation method in the design phase which simulates this standard RBAC definition method to which risk period information was directly added and a design refinement method which utilizes feedbacks from the evaluation results

1. はじめに

IT システムが企業活動の根幹となった現在、統制型セキュリティリスク対策は企業存続のために必須となっている。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一括管理がログ管理と合わせて中核技術とされている。その実現方式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イデンティティ・アクセス管理（IAM: Identity and Access Management）システムが実践されつつある。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のアクセス制御情報の一貫性制御が課題となっており、セキュリティ脅威、従業員の生産性と、システム構築/管理コストのトレードオフを考慮した方式が必要とされている。大企業に代表される広域分散システムでは、採用・退職、人事異動、設備の導入が発生した際、組織のポリシーに反しないことの確認を行い、影響するロール、権限を求め、その結果に影響するすべてのシステ

^{†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} 早稲田大学
Waseda University

^{†3} 京都大学
Kyoto University

^{†4} 三菱電機株式会社
Mitsubishi Electric Corporation

ムに配布して同期する必要がある。しかし、同期にはプラットフォームや運用等の制約による時間的遅延が発生するため、論理的にはポリシーに従った処理でも、過渡状態において一時的にポリシーに反する期間が生じる。権限付与遅延は権限保有者の機会損失を招くリスクが、権限抹消遅延は権限未保有者のアクセス違反を招くリスクが生じる。一方で、システム的にこの期間を短縮するためには、コストが発生する。このような対策として、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て、エンタプライズ RBAC^{4),5)}、アクセスポリシーをルールで表現するルールベース RBAC⁶⁾⁻⁸⁾ 等、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、特定の RBAC モデルを採用した個別システムでの性能評価⁹⁾にとどまっていた。そのため、多種多様な個別システムを広域ネットワークで接続した複合システムを評価して、与えられた要件を容認可能なコストで実現するシステムの設計支援を行うことは困難であった。

本論文では、このような異種 RBAC モデルを組み合わせた統合システムの総合的なリスク対策をシステム構築前の設計段階で評価する手法、および評価結果を設計にフィードバックする手法について提案する。最初に、RBAC の管理コマンドをセキュリティ対象への展開操作を含めて細分化し、その結果である部品を構成要素として UML¹⁰⁾ で定義する異種分散 RBAC システム構成モデルを示す。次に、このシステム構成モデルに、ネットワーク等のプラットフォーム情報、スケジューリング可能性、性能評価のプロファイル¹¹⁾⁻¹³⁾を付加して RBAC の動的振舞いを定義するリスク評価モデルを示す。さらに、実システムの性能評価結果を基本データとして与えて、RBAC 管理コマンドをシミュレートし、一時的にポリシーに反する過渡状態にある期間を評価する手法を示す。評価例として、監査目的でアイデンティティ情報、アクセス制御情報を世代管理するように拡張を施した RBAC モデル¹⁴⁾を採用した実システムを用いた。最後に、故障木を利用して異種分散環境システムの全体評価、および評価結果をフィードバックした設計手法について示す。2 章において、セキュリティポリシーに則ったアクセス制御情報設定の関連技術について示す。3 章では、異種分散 RBAC システム構成モデルと、実時間プロファイルを用いた異種分散 RBAC システム評価モデルについて示す。4 章では、世代管理 RBAC の実システム性能評価について示す。5 章では、全体評価を行う故障木を用いて、設計段階でのリスク評価、および評価結果をフィードバックした設計手法を示し、6 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、リスク分析で用いる関連概念について示す。

2.1 ロールベースアクセス制御 (RBAC) モデルとその拡張

RBAC モデル^{1),2)}では、ユーザ情報と、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 RBAC の標準である NIST (National Institute of Standards and Technology) のコア RBAC³⁾を、図 2 にその定義を示す。また、RBAC モデルの構成要素を維持管理する管理コマンドと、実行制御を行うシステム関数を図 3 に示す。コア RBAC では、ユーザとロールのマッピング関係を、ユーザとロールの対の集合 UA として定義しており、図 4 に示す $AssignUser(user, role)$ 、 $DeassignUser(user, role)$ 関数でその対を指定することに

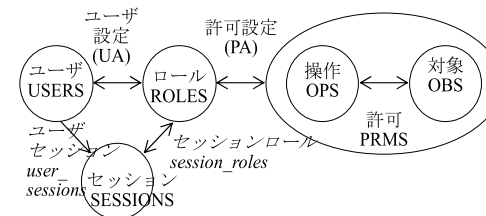


図 1 NIST コア RBAC
Fig. 1 NIST core RBAC.

定義 1. コア RBAC

- $USERS, ROLES, OPS, \text{ and } OBS$, 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.
- $assigned_users(r) = \{u \in USERS \mid (u, r) \in UA\}$, ロール r のユーザ集合へのマッピング.
- $PRMS = \mathcal{P}(OPS \times OBS)$, 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.
- $session_user(s:SESSIONS) \rightarrow USERS$, セッション s のユーザへのマッピング.
- $session_roles(s) \subseteq \{r \in ROLES \mid (session_user(s), r) \in UA\}$, セッション s のロールの集合へのマッピング.
- $avail_session_perms(s:SESSIONS) \rightarrow \mathcal{P}(PRMS)$, セッション s 内でユーザに有効な許可.

図 2 コア RBAC の定義

Fig. 2 Definition of core RBAC model.

(1) コア RBAC の管理コマンド

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

(2) コア RBAC のシステム関数

- $CreateSession(user:NAME; ars:2^{USERS}; session:NAME)$
- $DeleteSession(user, session:NAME)$
- $AddActiveRole(user, session, role:NAME)$
- $DropActiveRole(user, session, role:NAME)$
- $CheckAccess(session, operation, object:NAME; out result:BOOLEAN)$

図 3 コア RBAC の管理コマンドとシステム関数
Fig. 3 Administrative commands and system functions for core RBAC.

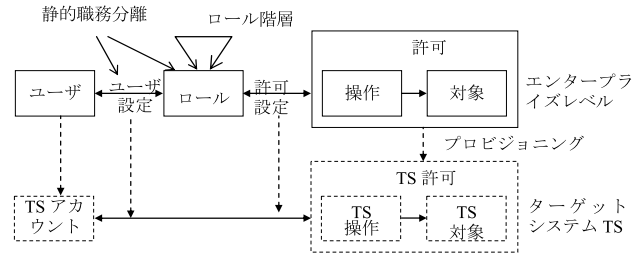


図 4 エンタープライズ RBAC モデル
Fig. 4 Enterprise RBAC model.

より、集合への追加、削除を行う。

文献 15) では、組織ロールとシステムロールを分離した複合 RBAC が提唱された。複合 RBAC では、ユーザー属性によるロールと、対象によるロールを分離することにより、大規模で複雑な組織にとって、スケーラブルで再利用可能な RBAC モデルを提供する。また、文献 16), 17) では、ユーザー設定、許可設定を決定する条件として、RBAC における主体ロールに加えて、対象ロール、環境ロールを導入し、それらを関連付けるアクセス仲介ルールを定めた一般化 RBAC (Generalized RBAC) が提唱された。

文献 4), 5) では、ターゲットシステムのユーザーとアクセス権を統合管理するシステムのモデルとして、エンタープライズ RBAC (ERBAC) モデルが提唱された。図 4 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た情報を、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ーを維持管理する。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ー設定 UA の定義には、通常、ルールが用いられる。文献 6) では、ユーザー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 が紹介された。ルールベース RBAC では、ユーザー属性の変更が生じて、 UA を変更する必要がないため、運用コストが低減される一方で、実行時の認可性能が課題となる。文献 7) では、図 5 に示すように、ユーザー-ロール設定を、エンタープライズレベルでは動的なルールで定義し、ターゲットシステムでは静的な設定とし、ルールを評価してプロビジョニングするモデルが提唱された。以下に示すように、ルールは、条件とアクションからなる。

```
IF User-CostCentre      = "AB2500" AND
   User-Company         = "Bank1"
THEN Assign Role "Bank1-Cashier"
```

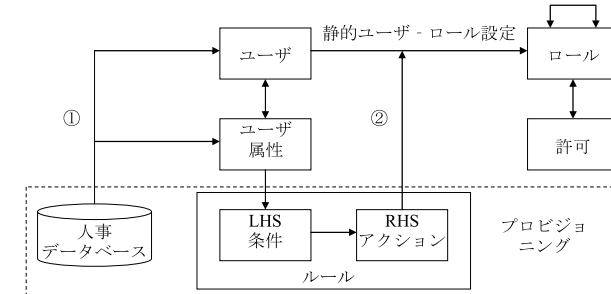
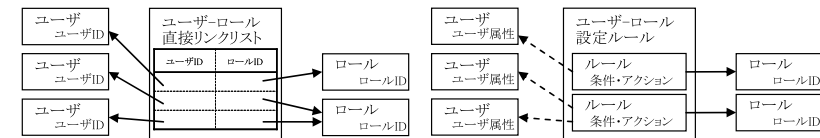


図 5 RBAC のルールベースプロビジョニング
Fig. 5 Rule-based provisioning of RBAC.



(a) 静的設定方式 (b) 動的ルールベース方式

図 6 ユーザー設定の実装方式

Fig. 6 Implementation of user assignment.

図 5 ① のプロビジョニングでは、ユーザーの追加削除、ユーザー属性の変更を行う。また、② のプロビジョニングでは、ユーザー属性に基づいて静的なユーザー-ロール設定、すなわち、図 2 に示すユーザーとロールの多対多のマッピング関係 UA に変換して追加削除を行う。

文献 14) では、ユーザーとロールの対を直接保持する静的設定方式とユーザーとロールの関係をルールで定義して実行時にユーザーとロールの対を求める動的ルールベース方式の 2 種類の RBAC モデルの実装方式の認証・認可の性能比較を行っている。図 6 に両者のユーザー設定の実装方式を示す。静的設定方式ではユーザーとロールの対のリストを、動的ルールベース方式ではユーザー設定を定義したルールを保持する。許可設定の実装方式についても同様に静的設定方式、もしくは動的ルールベース方式がとられる。

文献 8) では、ルールに基づく権利委譲とその取消しについて、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系統的に管理する手法が提唱された。しかし、これらは、手法の提案と定性的な評価までで、定量的なリスク評価、

設計への応用には触れていない。本論文では、RBAC モデルの異種分散性について、以下のように分類し、これらを定義可能なモデルを示す。

- (1) ポリシ設定方式：ユーザ設定，許可設定のポリシを直接リンクで設定する静的設定方式，およびルールで設定する動的ルールベース方式。
- (2) プロビジョニング元とプロビジョニング先のポリシ設定方式の組合せ：ERBAC における (1) のポリシ設定方式間の組合せ。
- (3) 異種ロールの組み込み：複合 RBAC，一般化 RBAC に代表される用途に応じた複数のロールの組み込み。
- (4) 大企業に代表される階層型エンタプライズシステム：ERBAC の構成を一般化した異種ターゲットシステムの分散型階層構成。

2.2 世代管理 RBAC モデル

RBAC モデルに基づいてアクセス制御を行うシステムにおいても，設定の誤り，権利保有者の悪意が原因で意図しない行為が行われることがある。そのための対策として，定期的な監査時，インシデント発生時に過去のログの解析が行われる。その際，ログとして残された行為と，行為が実行された時点のアクセス制御情報を照合する必要がある。文献 14) では，時刻印を用いたアイデンティティの世代管理手法とそれを利用したトラッキング手法が示されている。しかし，この文献では，世代管理に要するディスク容量の評価までで，過渡状態が生じる時間評価には触れられていない。

2.3 セキュリティポリシのコストとリスクの定量化，およびその課題

文献 18) では，RBAC の経済的インパクトとして，従業員あたりの効果を，情報システム部門の運用管理コスト低減，従業員生産性効果，セキュリティ効果の和として定量的に定義している。セキュリティ効果を具体的に調査した結果である文献 19)，20) を参照し，代表的な操作である，(1) 新ユーザへの既存権限設定，(2) 既存ユーザの権限変更，(3) 既存ユーザへの新権限作成，(4) 権限停止を対象とした RBAC と非 RBAC の定量的比較を行っている。しかし，これらは実システムの評価までで，RBAC モデルの異種性，評価結果を設計にフィードバックする手法については，言及していない。

2.4 UML とそれを利用した RBAC モデル定義

システム設計の手法として UML¹⁰⁾ 等によるモデリングが広く普及している。また，非機能要件であるスケジュール可能性・性能の解析ためのプロファイルが，主に組み込み系の分野で利用されている^{11)–13)}。このプロファイルでは，UML のコンポーネント図，クラス図，状態遷移図，シーケンス図等に，応答時間，スループット，期限，遅延時間，優先順位，

入力パターン/頻度等の時間制約を定義するためのステレオタイプ，タグの標準を定めている。また，入力パターンとして，周期（ジッタ付き），最小最大間隔，バースト等が用意されている。

文献 21) では，RBAC を題材にした UML メタモデルを定義し，セキュリティモデルからセキュリティアーキテクチャを自動生成するアプローチを提案している。文献 22) では，RBAC モデルを UML のクラス図で表現し，RCL (Role-based access Control Language) 制約を OCL (Object Constraint Language) 制約²³⁾ に変換して RBAC 実行コードを自動生成するモデル駆動開発アプローチを提案している。さらに文献 24) では，UML，OCL で定義したモデルをもとに RBAC ポリシの形式的な検証と UML を用いて設計したシステムのポリシの妥当性確認の試みを行っている。このように，これまでの研究では，権利の付与，抹消といった変更トランザクションがポリシに則していることの検証に UML を用いている。しかし，異種 RBAC モデル間のデータ同期，および過渡状態で一時的に発生するリスクの解析をモデル駆動のアプローチで扱っているものはない。本論文では，RBAC モデルに特化して形式的に定義したクラス図に，同期のための性能指標を付与した設計情報を用いて，設計段階におけるリスク評価，および評価結果をフィードバックした設計手法について示す。

3. 異種分散 RBAC モデル定義とそれを用いたリスク評価手法

本章では，分散環境において，RBAC モデル，およびその拡張モデルを採用するターゲットシステムのアイデンティティとアクセス権を統合管理するシステムのモデルである異種分散 RBAC モデルを定義する。また，それを用いたリスク評価手法を示す。

3.1 異種分散 RBAC モデルとそれを用いたリスク評価手法概要

本論文で提案する評価手法の概要を図 7 に，各ステップの内容を以下に示す。

- (1) システム構成設計プロセス
個別異種 RBAC モデルシステムとそれらのシステム間の構成を，クラス図を用いて設計
出力：異種分散 RBAC システム構成設計モデル
- (2) リスク評価設計プロセス
H/W，ネットワーク等のプラットフォームを定義した合成構成図，動的な振舞いを定義した状態遷移図，および処理負荷情報（処理時間，処理回数等），入力情報を定義した性能プロファイルを，システム構成モデルに付加

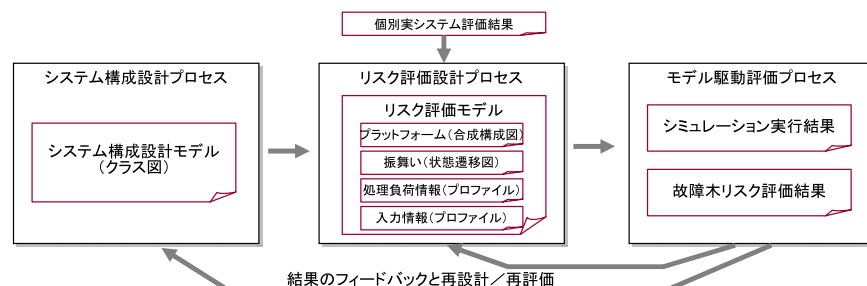


図 7 異種分散 RBAC モデル評価手法の概要

Fig. 7 Outline of heterogeneous distributed RBAC model evaluation method.

出力：異種分散 RBAC リスク評価モデル

(3) モデル駆動評価プロセス

(2)の結果であるモデル定義を直接実行したシミュレーション、および実行結果の評価をフィードバックした改良設計

出力：実行結果性能データ

故障木等を用いたリスク評価結果

本章では、(1)、(2)について示す。(3)については、4章、5章で示す。

3.2 異種分散 RBAC システム構成設計プロセス

3.2.1 モデル要件

エンタプライズレベルの RBAC モデルとして、2.1 節で示した ERABC モデルが提唱されている。しかし、RBAC モデルの管理コマンド、および、異なる RBAC モデルを採用したシステム間のプロビジョニングといった実装レベルの定義まで言及していない。本章では、以下の要件を満足するモデル定義を示す。

(1) 構成クラス

RBAC モデルを構成するユーザ、ユーザ設定、ロール、許可設定、許可をクラスとし、それぞれのインスタンスを操作する管理コマンドをメソッドとして持つ。

(2) ポリシ設定

ポリシをルールで設定する動的ルールベース方式と、インスタンスを直接リンクで設定する静的設定方式の2種類のポリシ設定方式の混在を許す。

(3) RBAC モデル間のプロビジョニング

(2)で示した2種類のポリシ設定方式のうち、同一方式を採用するシステム間、およ

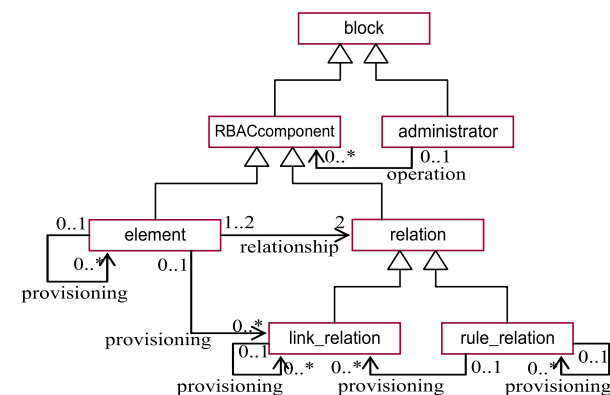


図 8 メタレベルの異種分散 RBAC モデル構成のクラス図

Fig. 8 Class diagram for meta-level heterogeneous distributed RBAC model.

び動的ルールベース方式を採用する上位レベルシステムから静的設定方式を採用する下位レベルシステムへのプロビジョニングを許す。

(4) ロールの複数段定義

2.1 節で示した複合 RBAC、一般化 RBAC を表現できるように、ユーザと許可の間に位置するロールを任意の段数定義可能とする。

(5) 階層型エンタプライズシステム

ERABC を拡張して、エンタプライズレベルシステムを起点に任意の数のレベルのターゲットシステムを階層的に配置したシステムで一元的にアクセス制御情報を管理する構成を定義可能とする。

(6) プロビジョニング時のロール評価

ERABC のターゲットシステムのように、ユーザと許可の直接関係を保持する場合に対応可能とするため、ロールを評価した直接関係のプロビジョニングを許す。

3.2.2 異種分散 RBAC システム構成設計モデルのクラス定義

3.2.1 項で示した要件を、クラス図を用いて形式的に定義したメタレベルの異種分散 RBAC システム構成設計モデルの定義を、図 8 に示す。本モデルは、クラス `block` とその関係からなる。`block` は以下よりなる。

- ▶ administrator：トランザクションを入力するシステム管理者。
- ▶ RBACComponent：RBAC モデルの構成要素とその関係。

▶ element: *USERS*, *ROLES*, *OPS*, *OBS* 等の RBAC モデルの構成要素.

▶ relation: *UA*, *PA* 等の $\ll\text{element}\gg$ 間のマッピング関係.

▶ rule_relation: 動的ルールベース方式に対応した $\ll\text{relation}\gg$.

▶ link_relation: 静的設定方式に対応した $\ll\text{relation}\gg$.

また, $\ll\text{block}\gg$ 間の接続は, 以下の 3 種類からなる.

▶ operation: データの挿入, 削除, 変更といった管理者からの操作.

▶ relationship: $\ll\text{element}\gg$ と $\ll\text{relation}\gg$ の関係. 具体的には, $\ll\text{relation}\gg$ からの $\ll\text{element}\gg$ の識別子, 属性値の参照.

▶ provisioning: 上位の $\ll\text{RBACcomponent}\gg$ から下位の $\ll\text{RBACcomponent}\gg$ へのプロビジョニング.

次に, メタレベルの異種分散 RBAC システム構成設計モデルから, システムレベルのモデルを定義する. クラス間の接続は, 以下の部品を用いて定義する.

- ポート: 他の構成要素と情報を授受する口
- インタフェース: 他の構成要素に提供/要求するサービス
- コネクタ: 構成要素のポート間のリンク

図 9 (a) に動的ルールベース方式を採用したエンタプライズシステムと静的設定方式を採用したターゲットシステム TS1 から構成されるシステムの構成設計モデルを示す. エンタプライズレベルシステム管理者から管理コマンドが起動されると, エンタプライズシステムに対する変更情報を評価して全構成要素をターゲットシステムにプロビジョニングして同期をとることを表す. 図 9 (b) にエンタプライズレベルのロールを省略して, ユーザと許可の直接関係をターゲットシステム TS2 にプロビジョニングする場合を示す. エンタプライズレベルで更新が生じると, 以下に示すユーザと許可の関係 *UP* を求めてプロビジョニングする.

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

このように異種 RBAC モデルの全構成要素を網羅的に配置したシステム構成設計モデルをテンプレートとして, 対象とするシステムを構成する要素を選択して設計する.

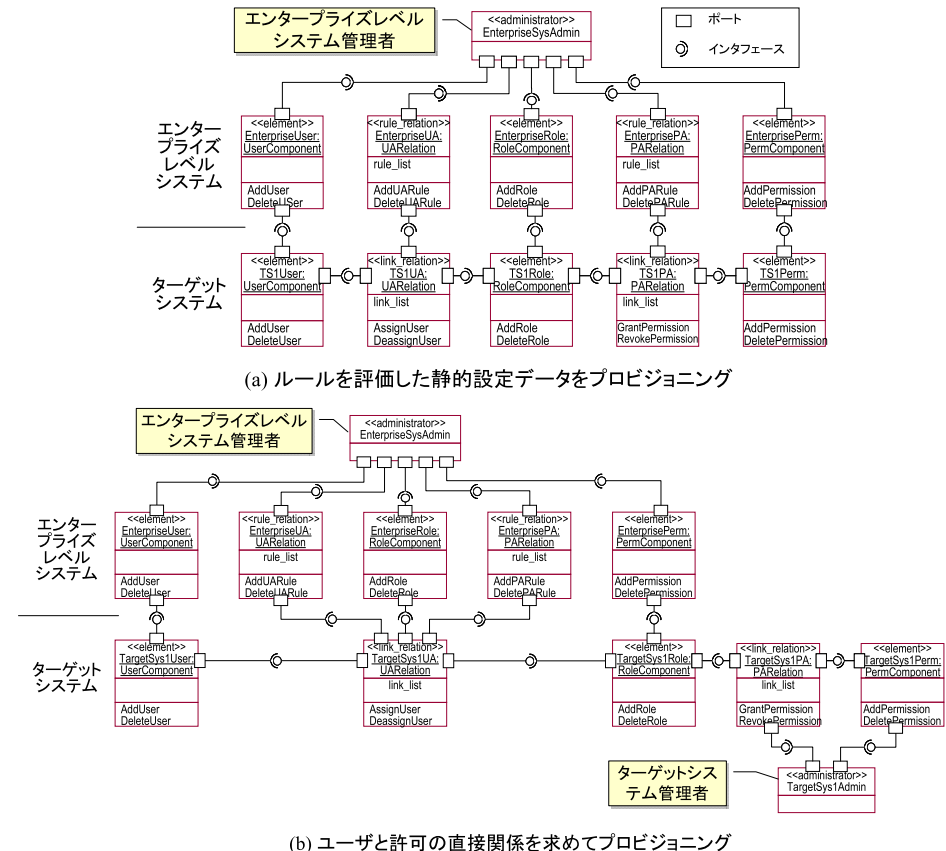


図 9 異種分散 RBAC システム構成設計モデル

Fig. 9 System architecture design model for heterogeneous distributed RBAC.

3.3 動的情報を付加した異種分散 RBAC リスク評価プロセス

本節では, 異種分散 RBAC システム構成設計モデルに動的情報等を付加したリスク評価モデルを定義し, 過渡状態にあるためにリスクが生じる期間を設計段階で評価するための手法について示す.

3.3.1 リスク評価プロセスの全体構成

プロビジョニングによって発生する過渡状態を明に表現するために, 3.2 節で示したシス

表 1 過渡状態におけるリスク
Table 1 Risks in transient states.

	管理コマンド	$t_i \leq t_1 < t_2$ 事後投入	$t_1 < t_i < t_2$ 事前投入	$t_1 < t_2 \leq t_i$ 事前投入
権利付与	<i>AddUser, AddRole,</i> <i>AssignUser,</i> <i>GrantPermission</i>	機会損失リスク $t_2 - t_i = t_1 + pr - t_i$	機会損失リスク $t_2 - t_i = t_1 + pr - t_i$	セキュリティ リスク $t_i - t_2 = t_i - t_1 - pr$
権利抹消	<i>DeleteUser, DeleteRole</i> <i>DeassignUser,</i> <i>RevokePermission</i>	セキュリティ リスク $t_2 - t_i = t_1 + pr - t_i$	セキュリティ リスク $t_2 - t_i = t_1 + pr - t_i$	機会損失リスク $t_i - t_2 = t_i - t_1 - pr$

テム構成設計モデル定義に、エンタプライズレベルシステムの機能であるターゲットシステムへの構成要素のプロビジョニング機能を独立させたプロビジョニングシステムを設けることとする。プロビジョニングは、予約情報を用いた事前投入、または、発生した変更情報を用いた事後投入のいずれかで実行されるものとする。

エンタプライズレベルシステムへの変更トランザクション開始時刻を t_1 、ターゲットシステムへの反映時刻を t_2 、処理時間を $pr = t_2 - t_1$ 、適用の理想時刻を t_i とした場合の機会損失リスク、セキュリティリスクとその発生期間を表 1 に示す。処理時間 $pr > 0$ であること、および身分証明紛失等予測不可能なイベントによる事後投入を考慮すると、機会損失リスク、セキュリティリスクをゼロにすることは現実的ではない。そこで、一般的にはシステムとしては許容範囲を設けて運用で補う措置がとられる。システム設計の際、許容範囲内とするためには以下の手法が用いられる。

- (1) 処理時間 $pr = t_2 - t_1$ の短縮
- (1-1) プラットフォームの性能向上による処理時間の短縮
- (1-2) 採用する RBAC モデルの変更等、処理内容見直しによる処理時間の短縮
- (2) 各リスクのトレードオフを考慮した開始時刻 t_1 の設定
- 本章では、(1) に注目して、設計段階で許容範囲内の検証を行う手法を示す。(2) については、4 章で述べる。

3.3.2 RBAC モデル管理コマンドにおける評価対象と評価手順

3.2 節で示したシステム構成設計モデルにおいて RBAC モデルの管理コマンドのリスクが生じる期間にある確率を求める。管理コマンド i の変更トランザクション ij が、単位時間内に n 個、発生すると仮定する。このとき、リスクが生じている確率、すなわち、単位時間内でリスクが生じている時間の割合 p_i を以下の式で定義する。

$$p_i = \frac{\sum_{j=1}^n (e_{ij} - s_{ij}) \times u_{ij}}{T \times U} \quad (1)$$

s_{ij} : トランザクション ij の投入時刻
 e_{ij} : トランザクション ij の終了時刻
 u_{ij} : トランザクション ij の対象総人数
 n : トランザクション数
 U : 操作対象の総人数
 T : 対象時間

この式の変数である s_{ij} と e_{ij} を求める必要がある。これらの値は、排他制御、処理時間、運用スケジュールに依存する。特に、採用する RBAC モデルの差異、管理コマンド、プロビジョニング方式を考慮した運用設計と処理時間の測定が課題となる。近年、上流工程の設計情報を利用するモデル駆動アーキテクチャが実用化されつつあり、特に、実時間システムを対象とした解析の標準化が進められている。本章では、スケジュール可能性、性能、時間解析のための実時間プロファイル^{11)–13)}を参考にして新たに RBAC モデル向けに定義したモデル駆動性能評価手法について示す。3.2 節で定義した RBAC モデルに特化して詳細化した異種分散 RBAC システム構成設計モデルにおいて、管理コマンド実行時の個々の構成要素の動的な振舞いを表現し、かつ、リスクが生じる期間の時間解析が可能な評価手法を提案する。図 10 に本手法で用いる評価システムの全体構成を示す。以下に本システムを用いた評価手法を示す。

[入力]

- 異種分散 RBAC リスク評価モデル定義：3.2 節で定義した異種分散 RBAC システム構成設計モデルに以下の要素を付加したモデル定義。
 - ▶ プラットフォームを定義する合成構成図。
 - ▶ 管理コマンド実行時の動的な振舞いを定義する状態遷移図。
 - ▶ 処理負荷情報、入力情報を定義する実時間プロファイル。

[出力]

- イベントごとの開始時刻 s_{ij} 、終了時刻 e_{ij} 。

[手順]

- ① トランザクションを運用で定められたスケジュールで待ち行列に置く。
- ② 各クラスは待ち行列からイベントを取り出して、状態遷移図、処理負荷情報、入力情報

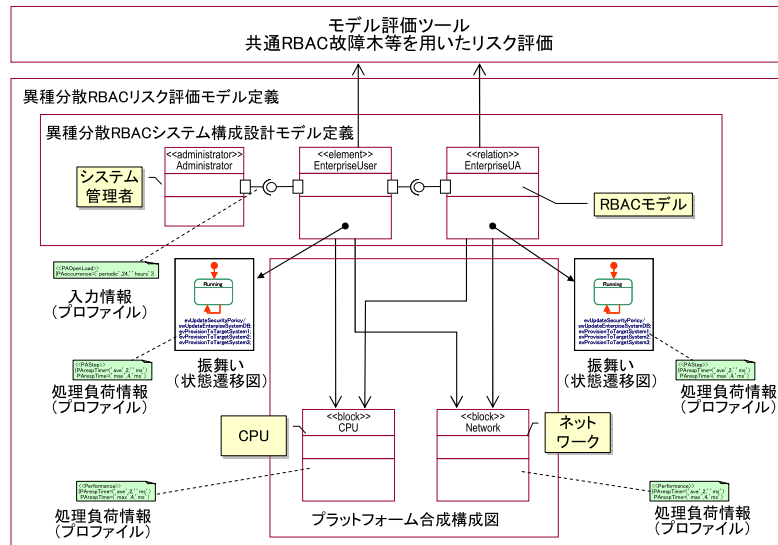


図 10 評価システムの全体構成
Fig. 10 Structure of evaluation system.

を示すプロファイルに沿って状態を変化させ、次イベント発生を行う。

- ③ 状態遷移の処理負荷情報を示すプロファイルとプラットフォームの性能情報の積を処理時間として状態遷移の時刻をモデル評価ツールに出力する。
- ④ すべての入力列が完了するまで、①～③を繰り返す。

入力の詳細については、3.3.3 項で示す。本評価システムは以下の特長を持つ。

- システム設計で作成したモデルに、プロファイルを用いて性能評価のための時間情報を直接記述するので、設計するシステムとの矛盾が生じない。また、待ち行列モデル等付随するモデルの作成が不要である。
- 異種分散 RBAC リスク評価モデルを入力として、シミュレーション用のコードを作成・実行するので、モデルレベルでのシミュレーションが可能である。
- プラットフォームに関する記述を設計情報と独立して指定可能としたので、入力する関数群のスケジューリング、CPU 性能、サーバ台数といったチューニングを独立して行うことが可能である。その際、設計情報の変更は不要である。

```

TimeValue ::= <time-value>
RTArrivalPattern ::= <bounded-string> | <bursty-string> | <irregular-string> | <periodic-string>
<time-value> ::= ( integer, <time-unit> )
<time-unit> ::= " ms " | " s " | " min " | " hr " | " days "
<bounded-string> ::= " bounded " , " <time-value> " ,
    <time-value>
<bursty-string> ::= " bursty " , " <time-value> " , " <integer> "
<irregular-string> ::= " irregular " , " <time-value> [ " , "
    <time-value> ] *
<periodic-string> ::= " periodic " , " <time-value> [ " , "
    <time-value> ]
  
```

図 11 処理負荷情報と入力情報のスケジュールの定義

Fig. 11 Definition of performance information and input schedule.

3.3.3 異種分散 RBAC リスク評価モデル定義

異種分散 RBAC リスク評価モデル定義の構成を以下に示す。

- (1) 異種分散 RBAC システム構成設計モデル (クラス図)
- (2) プラットフォーム (クラス図)
異種分散 RBAC システムを実行させる H/W、ネットワークといったプラットフォームを定義する。
- (3) 管理コマンド実行時の動的な振舞い (状態遷移図)
管理コマンド実行時の構成要素の振舞いを定義する。構成要素の状態/条件変化のトリガであるイベントによって遷移する。
 - 状態：構成要素の動作が異なる状況を区別したもの
 - 状態遷移：イベントの発生により引き起こされる状態の変化
 - アクションリスト：状態遷移にともなう構成要素の動作
- (4) 処理負荷情報、入力情報 (プロファイル)

2.4 節で示した UML モデリングの実時間プロファイルを用いて、状態遷移に要する処理負荷情報と入力情報のスケジュールを定義する。処理負荷情報は、図 11 に示す TimeValue で指定する。入力情報は、ステレオタイプの RTArrivalPattern を用いて最小最大間隔 (bounded)、バースト (bursty)、非定期 (irregular)、ジッタ付き周期 (periodic) を指定する。図 9 (a) の一般的なターゲットシステム ts1 に対応した定義例を図 12 に、図 9 (b) のユーザと許可の関係 UP のプロビジョニング、ターゲットシステム固有のアクセス権設定を加えたターゲットシステム ts2 に対応した定義例を図 13 に示す。いずれも、プラットフォームは省略した。

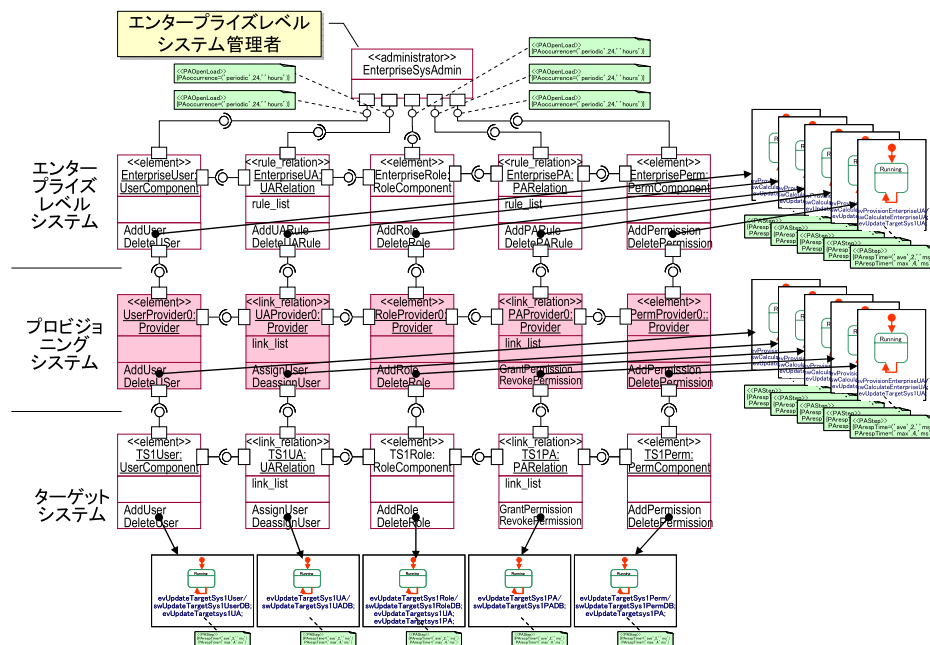


図 12 一般化異種分散 RBAC リスク評価モデル

Fig. 12 General heterogeneous distributed RBAC risk evaluation model.

3.4 考 察

3.2 節, 3.3 節で, 2.1 節で示した RBAC の代表的な拡張モデルが, 異種分散環境で定義可能なことを示した. また, セキュリティポリシとして上位レベルで定義されたルールの解釈は, (a) エンタープライズレベルシステム入力前, (b) エンタープライズレベルシステムのデータベースへの登録時点, (c) プロビジョニング時点, (d) ターゲットシステムがルールを受信した時点, (e) ターゲットシステムにおける認可時点のいずれかで行われて, 直接リンクに変換される. 本モデルでは, 表 2 に示すように, 各システムの <<relation>> の属性値とルール解釈による遅延を指定する状態遷移図の箇所を定義することによって (a)~(e) のすべての場合を指定することが可能である.

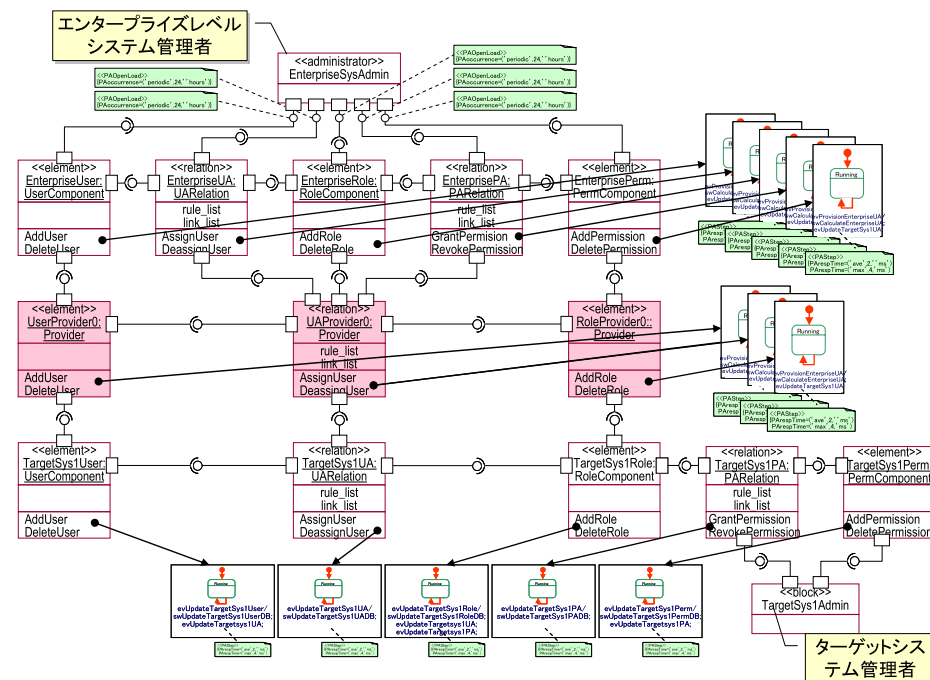


図 13 プロビジョニング時解釈異種分散 RBAC リスク評価モデル

Fig. 13 Heterogeneous distributed RBAC risk evaluation model for rule estimation in provisioning.

表 2 ユーザ設定 UA の指定方法

Table 2 Specification of user assignment UA.

指定箇所	(a)	(b)	(c)	(d)	(e)
エンタープライズレベルシステム属性,	link list	link list	rule list	rule list	rule list
ターゲットシステム属性	link list	link list	link list	rule list	rule list
遅延を指定する状態遷移図	システム外 (事前)	エンタープライズレベル	プロビジョニング	ターゲット	システム外 (実行維持)

ユーザ(USERS)				組織(ORGS)			
BeginTS	EndTS	UserID	Attributes	BeginTS	EndTS	OrgID	Attributes
201004010000	201006300000	ユーザ A	資材 1 課, 課長	201004010000	<i>null_{max}</i>	資材 1 課	資材部
201006010000	<i>null_{max}</i>	ユーザ B	人事 1 課, 担当	201004010000	<i>null_{max}</i>	人事 1 課	人事部

ユーザ設定(UA)			
BeginTS	EndTS	RoleID	UA
201004010000	<i>null_{max}</i>	ロール 1	所属組織:資材部 ∧ 役職:課長
201004010000	<i>null_{max}</i>	ロール 2	所属:人事部
201007010000	<i>null_{max}</i>	ロール 3	役職:課長

許可設定(PA)				
BeginTS	EndTS	ObjID	OPS	PA
201004010000	<i>null_{max}</i>	扉 1	開錠	ロール 1
201004010000	200906300000	人事システム	起動	ロール 2
201007010000	<i>null_{max}</i>	人事システム	起動	ロール 2 ∨ ロール 3

図 14 ルールベース階層組織 RBAC の世代管理例

Fig. 14 Examples of multi-version management for rule-based hierarchical organization RBAC.

4. 世代管理 RBAC システムの基本データ採取とモデル駆動評価プロセス

本章では、3 章で定義した RBAC モデル定義にプラットフォーム定義を加えたモデルを用いて、実システムである世代管理 RBAC システム^{14),25)}を対象にしたモデル駆動評価プロセスについて示す。本プロセスでは、既存のモデル実行エンジンに新たに開発した性能指標値計算機能、性能指標値検証機能を加えて実行結果をモニタリングし、性能指標値検証を行った。本論文ではモデル実行ツールとしてモデル駆動開発ツール IBM/Telelogic Rhapsody Developer Edition 7.2 を利用したが、他の同等のツールでも代替が可能である。

4.1 世代管理 RBAC システム

2.2 節で示した世代管理 RBAC システム¹⁴⁾では、組織階層をロールと独立して管理するルールベース組織階層 RBAC を採用している。図 14 にルールベース組織階層 RBAC の各構成要素に対応した世代の管理例を示す。これらのテーブルは、値の有効期間として、開始時刻印 BeginTS と終了時刻印 EndTS を持つ。レコードの追加、削除時には、更新前のレコードを完全消去せず、EndTS の値を変更して削除状態とする。なお、期限未定値として、すべての時刻印 t_i について、 $t_i < null_{max}$ となる *null* 値を用意し、最新レコードを示すこととした。文献 14) では、従業員 5,000 名程度の企業を対象に、本 RBAC 拡張モデルの世代データ管理に要するディスク容量の評価を行った。本論文では、プロビジョニングに要する時間で発生するリスク評価を行うこととする。

4.2 世代管理 RBAC システムの性能評価

世代管理 RBAC モデルをエンタプライズレベルのアイデンティティ管理システムとして採用するためには、定期的な人事異動時や、データの棚卸し時の、大量バッチによるデータ

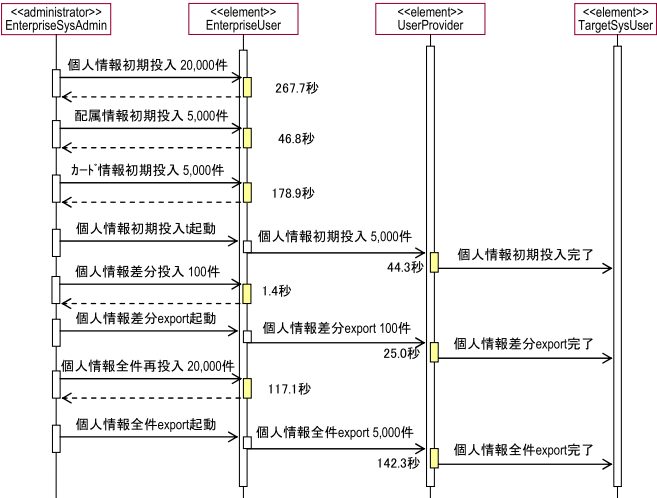


図 15 世代管理 RBAC システムの性能評価

Fig. 15 Measurement results of multi-version RBAC.

投入、プロビジョニングデータ作成性能が課題となる。4.1 節で示したモデルを採用した実システムで、以下の従業員数を対象に、性能測定を実施した。

- 従業員：20,000 人
- ターゲットシステム利用ロール該当者：5,000 人
- アイデンティティ管理システムの構成
 - CPU：Intel(R) Core™2 Duo CPU T9400 @2.53 GHz、メモリ：2.00 GB
 - OS：Windows Vista Ultimate Sp2, M/W：Tomcat 5.5.27, Oracle 10g R2

図 15 に、データ投入、プロビジョニングシナリオに沿って測定した実測値を示す。また 20,000 件のユーザを 7 世代蓄積した場合の性能差は、全 20,000 件再投入（更新）で 102.8～115.7 秒、5,000 件プロビジョニングで 142.3～157.9 秒と 13%以内であった。全件再投入を行った場合、投入前の値がすべて保存されるが、全件再投入が必要となる大幅な人事異動は、通常 2 回/年程度であることを考慮すると、世代管理による容量拡大が全体性能に及ぼす影響は容認範囲内であると考えられる。次節以降で、本測定結果を、モデル駆動性能評価の基礎データとして利用することとする。

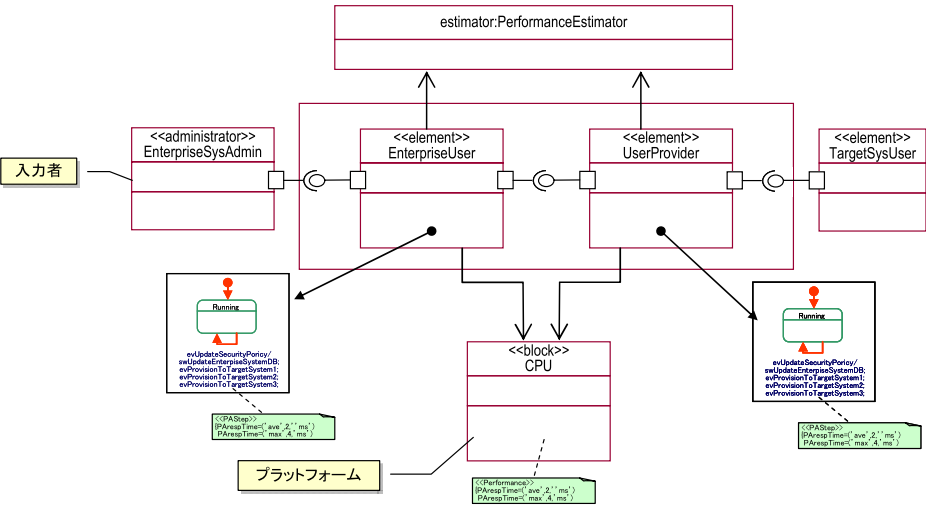


図 16 データ投入クラス図
Fig.16 Class diagram of data entry.

4.3 世代管理 RBAC システムモデル駆動解析

3 章で示した手順に従って性能評価シミュレーションを行う。性能指標値として、システム応答時間、システムスループット、コンポーネント利用率を求める。さらに、プラットフォームの CPU、ネットワークの指標値を変動させて、スケジュール可能性の分析、パラメータ調整を行う。ここでは、簡単のため、3.2 節で示した異種分散 RBAC システム構成設計モデルのうち、データ投入、ロール解釈とプロビジョニングデータ生成を対象に評価を行った例を示す。図 16 にデータ投入クラス図を示す。図中の EnterpriseUser が、エンタプライズレベルのユーザのクラスに、UserProvider が、ユーザのプロビジョニングのクラスに対応し、EnterpriseUser でデータ投入を、UserProvider でプロビジョニングデータ生成を行う。処理は排他で、各クラスの入力ポートは待ち行列を持つ。以下に示すシナリオに従って評価を行う。

- (a) 人事情報定期入力：社員情報を AddUser コマンドで EnterpriseUser に定期投入し、UserProvider が連続してプロビジョニング。
- (b) 個人情報ランダム入力：社員外情報を AddUser コマンドで EnterpriseUser にランダム投入し、結果を UserProvider にて一時的に蓄積し、定期的にターゲットシステムに

ID	種別	イベント	内容	経過時刻(秒)	処理時間(秒)
1	input	evSimUpdateUserAll	全件投入起動	0.0	
1	output	evSimUpdateUserAllDone	全件投入完了	259.4	259.4
2	input	evSimUpdateUserDiff	差分投入起動	129,600.0	
2	output	evSimUpdateUserDiffDone	差分投入完了	129,626.4	26.4
3	input	evSimUpdateUserDiff	差分投入起動	259,200.0	
3	output	evSimUpdateUserDiff	差分投入完了	259,226.4	26.4

(a) 人事情報定期入力結果
Measurement results of periodic input of personnel data

構成	% 1 回/3 時間一括プロビジョニング、1 日間 105 トランザクション投入
- シミュレーション全時間	= 86675.00 秒
性能推定値	
- システム性能	
応答時間	平均 = 5661.529 秒
	標準偏差 = 3201.629 秒
構成	% 1 回/1 日一括プロビジョニング、8 日間 101 トランザクション投入
- シミュレーション時間	= 691475.00 秒
性能推定値	
- システム性能	
応答時間	平均 = 41865.964 秒
	標準偏差 = 26065.577 秒

(b) 個人情報ランダム入力結果
Measurement results of random input of personal data

図 17 モデルシミュレーション結果
Fig.17 Model simulation results.

プロビジョニング

(a), (b) の基礎データは、4.2 節で実測した値である全件ユーザ投入 117.1 秒、プロビジョニング 142.3 秒と、差分ユーザ投入 1.4 秒、プロビジョニング 25.0 秒を設定した。上記、(a), (b) について IBM/Telelogic Rhapsody Developer Edition 7.2 上でシミュレーションを行い、 s_j と e_j を測定した結果を図 17 に示す。

(a) は、定期的な人事異動を想定し、社員情報の半年に 1 度の全件投入、半月に 1 度の差分投入を想定した。排他制御による待ち時間は発生しないので、4.2 節の実測値である全件投入の合計時間 $117.1 + 142.3 = 259.4$ 秒と、差分投入の合計時間 $1.4 + 25.0 = 26.4$ 秒が、シミュレーション結果となる。

(b) では EnterpriseUser に社員外情報をランダム入力し、UserProvider は、一括して処理を行う場合の平均処理時間をシミュレーションで求めた。EnterpriseUser, UserProvider

の処理時間の単純な合計値ではなく、ランダム入力後の EnterpriseUser での即時処理完了から一括処理までの差分、処理中の排他制御によって発生する待ち時間が加えられる。これらのデータを、3.3.2 項の式 (1) に代入して、RBAC モデル管理コマンドであるユーザ投入 *AddUser* のリスクの確率 $p_{AddUser}$ を求める。式 (1) では、分母が対象とするのべ時間数を、分子がリスクとなるのべ時間数を示す。シミュレーションではリスクが生じている平均時間数を求めているので確率を求めるうえでは、操作対象の総人数 U が相殺される。以下は、データ投入日 (1 日) のリスクの確率を示す。この結果から、投入から反映までの時間が、一括処理の時間間隔に依存することが確認される。

$$\text{一括処理 1 回/3 時間の確率 } p_{AddUser} = \frac{5,661.529 \times U}{86,675.00 \times U} = 0.0653 \quad (2)$$

$$\text{一括処理 1 回/24 時間の確率 } p_{AddUser} = \frac{41,865.964 \times U}{(691,475.00/8) \times U} = 0.4844 \quad (3)$$

また、式 (2) に示すように 3 時間ごとの一括処理のシミュレーションでは、排他制御による待ち時間が観測されている。一方で、1 日ごとの一括処理のシミュレーションでは、待ち時間が観測されていないので、式 (3) に示す一括処理 1 回/24 時間を 3 時間に補正した値である $0.4844/8 = 0.0606$ と、式 (2) に示す一括処理 1 回/3 時間の確率 0.0653 と比較すると、待ち時間の分、リスクが増大していることが測定結果から確認することができる。

次に、必要に応じてプラットフォームに与える性能情報を利用したチューニングを行う。3.2.2 項の手順③で示したように、状態遷移の処理負荷情報を示すプロファイルとプラットフォームの性能情報の積を処理時間として状態遷移の時刻をモデル評価ツールに出力する。たとえば、図 17 (b) の 3 時間ごとの一括処理の場合、H/W 性能を 0.5 と設定すると、EnterpriseUser の処理時間が 0.7 秒、UserProvider の処理時間が 12.5 秒と自動変換されて、シミュレーションを実行する。EnterpriseUser の処理時間向上は、3 時間ごとの処理の待ち時間に吸収されるので、応答時間は、UserProvider が改善され、平均応答時間 5,583.404 秒、確率 0.0644 の結果を得ることができる。

4.4 考 察

4.3 節では RBAC モデルのユーザ *User*、ユーザの更新管理コマンド *AddUser* に注目してシミュレーションを行った。この結果、ユーザの更新に関する RBAC モデル定義に、実システムを用いた基礎データを設定してシミュレーションすることにより、セキュリティリスク、機会損失の原因となるデータ更新の遅延時間を、定量的に測定できることを示した。以上の定量的データから、以下に示す特長を有する分析を行うことが可能となる。

- 世代管理 RBAC モデルの管理コマンドユーザ投入実装例をもとに、シミュレーション例を示した。同様にして、他の管理コマンド、RBAC 拡張モデル、データ投入方式に適用することが可能である。
- 結果をフィードバックしてプラットフォームの改善、別マシンでデータ準備を行う切替え、事前投入等の方策をリスク評価モデル上で再定義し、シミュレーションして比較することが可能である。
- 許容範囲の概念を導入し、上記時間から除いた比較が可能である。たとえば、1 日の反映遅延が許容範囲として認められる場合は、上記結果はシステム運用管理コストが機会損失として加算されない。特に半年に 1 回の全件棚卸しを、休日、夜間の利用といった運用によって回避される。このような許容範囲に収める方策は、RBAC モデルのシミュレーションでスケジュール可能性解析により評価することができる。
- 権限の停止 (退職者、認証用 IC カード紛失等) といったセキュリティ違反を招く事象をバーストとして計測して、即時反映するための性能解析が可能である。

4.5 モデル評価

本論文で提案するモデルでは、RBAC モデル実装の設計支援を目的に、シミュレーション結果を設計にフィードバックするために適切な粒度での性能データ収集と、性能評価モデルのためのシミュレーション部品の提供による設計コストの低減を図った。本節では、これらの観点で、提案モデルの評価を行う。

4.5.1 シミュレーションの粒度の要件

3.2 節で示した RBAC モデル実装の要件に則した設計とその妥当性評価を行うためには、表 3 に示す粒度まで詳細化した性能データ収集、パラメータ設定と、シミュレーション結果をフィードバックした再設計のサイクルが必要となる。(1) RBAC 管理コマンド単位、(2) 構成要素単位まで RBAC モデルを詳細化することによって、問題箇所を特定したデータ構造、採用する RBAC モデル、ルールの評価場所等の選択が可能となる。また、(3) RBAC モデルの異種性、(4) 構成要素への入力方式、(5) 構成要素間の同期方式、(6) プラットフォーム構成要素単位の性能までパラメータ設定を詳細化することによって、システムアーキテクチャの再設計時の方式選択が可能となる。

これまでの研究¹⁸⁾⁻²⁰⁾では、実システムをブラックボックスとして扱い、(1) の観点、すなわち、新ユーザへの権限設定、権限変更、新権限設定、権限停止といった単位での性能評価までで、設計段階で必要とされる粒度での性能データ測定まで至っていない。したがって、ボトルネックとなる箇所の特定、設計データへの反映を行うことが困難であった。4.3

表 3 シミュレーションの粒度
Table 3 Granularity of simulation.

粒度軸	詳細設計に必要とされる粒度の要件	4.3 節のシミュレーション例
(1) RBAC 管理コマンド単位の性能データ	図 3 示した RBAC モデルの各管理コマンド単位の性能データ収集	管理コマンド <i>AddUser</i> に特定した性能データを収集した。
(2) 構成要素単位の性能データ	RBAC モデルを構成するユーザ、ロール、許可、エンタープライズシステムレベル、プロビジョニングシステムレベル、ターゲットシステムレベルの各単位の性能データ収集	RBAC モデルを構成するエンタープライズシステムレベルユーザ、プロビジョニングシステムレベルユーザ単位での性能データを収集した。
(3) RBAC モデルの異種性の設定	2.1 節で示したコア RBAC、ルールベース RBAC、一般化 RBAC、ERBAC といった異なる RBAC モデルの指定とその組み合わせた設定	エンタープライズシステムレベルはルールベース RBAC、プロビジョニングシステムレベルはコア RBAC という組み合わせの設定を可能にした。
(4) 構成要素への入力方式の設定	ランダム投入に対するイベント処理、定期バッチ等による周期処理を構成要素ごとに指定した設定	エンタープライズシステムレベルはランダム投入によるイベント処理、プロビジョニングシステムレベルは周期処理といった混在を含む設定を可能にした。
(5) 構成要素間の同期方式の設定	排他制御による待ち行列、イベント処理結果を蓄積して夜間等に周期処理といった要素間の同期処理の設定	(4)の設定に連動して、待ち行列処理、イベント処理結果を蓄積して夜間等に周期処理といった要素間の同期処理の設定を可能にした。
(6) プラットフォーム構成要素単位の性能の設定	CPU、ネットワーク等のプラットフォーム構成要素単位の性能の設定	システム外初期設定値の相対値による CPU 性能の設定を可能にした。

節で示したシミュレーションにおける性能データ収集、およびパラメータ設定を、表 3 の“4.3 節のシミュレーション例”欄に記載した。これらは、本シミュレーションにおいて粒度要件に則した性能データ収集とパラメータ設定が実現されていることを示している。

4.5.2 設計コストの低減

4.5.1 項で示した要件を満足するシミュレーションは、一般的な分散システムでの処理時間のシミュレーションの手法を用いて実現することは可能であると考えられる。しかし、本論文で提案するモデルが提供する部品を利用することにより、設計コストの低減、誤りの混入の回避を図ることが可能となる。本論文の提案モデルでは、以下に示すシミュレーション部品を提供する。各部品を利用した効果を追記して示す。

- (1) RBAC モデルシステム構成設計モデルテンプレート
RBAC モデルの異種性を吸収した要素単位まで細分化したシステム構成設計モデル（クラス図）をテンプレートして提供し、クラスおよびインタフェースの組合せで設計可能とした。
- (2) 同期処理のための部品

一般の分散システムの処理時間のシミュレーションモデルでは、イベント処理に対する待ち行列は用意されており組み込んで利用することができる。しかし、分散アイデンティティ管理で必要とされる、入力イベント処理の実施後の結果であるイベントを次段階で周期処理として実施するための機構は用意されておらず、新たに部品として提供した。

- (3) 独立したプラットフォームテンプレート
CPU、ネットワークといったプラットフォームを独立させたテンプレートを用意することにより、サーバ構成、性能チューニングをパラメータ設定のみで可能とした。
- (4) モデル評価手法プロセス
3.1 節の図 7 で示したように、システム構成設計プロセス、リスク評価設計プロセス、モデル駆動評価プロセスでモデルを共有する評価手法を提供することにより、相互変換による誤りの混入を防ぐことを可能とした。

この結果、設計者は、以下に示すシミュレーションパラメータの設定作業を行うのみで、4.5.1 項に示した要件を満足したシミュレーションが可能となる。

- (1) RBAC 方式の選択
モデルシステム構成設計モデルテンプレートをもとに、RBAC モデルの選択、ロール段数、階層レベル数の設定を行う。
- (2) プラットフォームの選択
(1) で選択した構成要素のサーバ配置、ネットワーク構成を指定する。
- (3) 性能データの設定
3.1 節の図 7 における処理負荷情報（プロファイル）、入力情報（プロファイル）を設定する。数値データの設定手法については、5 章で詳述する。

5. 故障木を用いた RBAC モデル適用システムリスク評価方式へのモデル駆動解析結果の適用

本章では、4 章で示したシミュレーション結果を、文献 26)、27) の故障木を用いた RBAC モデル適用システムのリスク評価へ適用して、設計段階におけるリスク評価、および評価結果をフィードバックした設計手法について示す。

5.1 故障木を用いた RBAC モデル適用システムリスク評価方式概要

文献 26)、27) では、図 18 に示した故障木を用いた異種 RBAC モデル適用システムのリスク評価方式を提唱している。本故障木は、2.3 節で示したセキュリティポリシのコスト

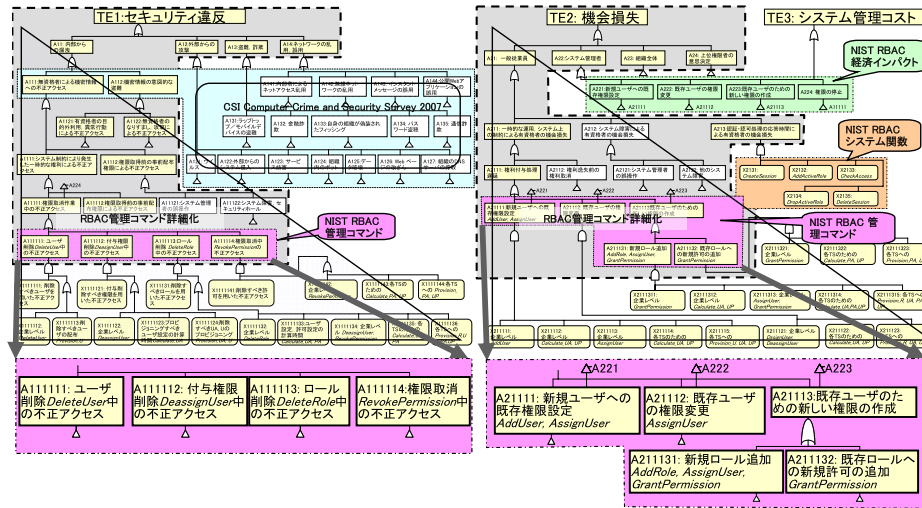


図 18 共通 RBAC 故障木の例

Fig. 18 Example of common RBAC fault trees.

とリスクの定量化をもとに“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，RBAC モデルの管理コマンド，システム関数単位まで詳細化して，異種 RBAC モデルを組み合わせた統合システムの総合的なリスク評価を可能とした。しかし，本手法では，性能測定が可能な実システム評価を対象としており，設計段階での評価，および評価結果をフィードバックした設計手法については，言及していない。本章では，以下の2点を付加することにより，それらを可能とした設計手法について示す。

- 故障木によるリスク評価を，3章で示した異種分散 RBAC モデル評価手法においてモデル駆動評価プロセスの後半に位置づけて，シミュレーション結果を故障木解析に入力し，評価結果をリスク評価設計プロセス，システム構成設計プロセスにフィードバックして設計支援を行う。
- ポリシーの許容範囲の概念を導入して，フィードバックの対象，目標を明確化する。

5.2 評価結果を利用した設計手法

3.3節で示したように，セキュリティリスク，機会損失リスクをゼロにすることは現実的ではないので，許容範囲を設けた運用を行うことが一般的である。一方，処理開始時刻，終了時刻は，データソースとなるシステムのデータ準備や，ターゲットシステムの稼働状況等

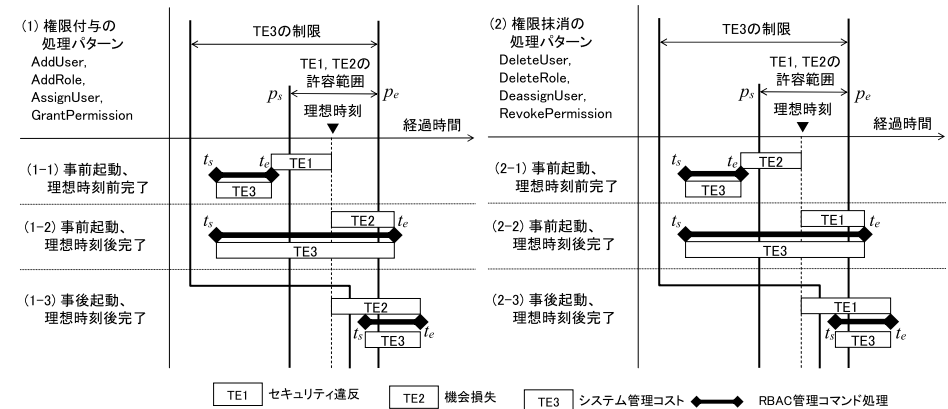


図 19 トップ事象と RBAC 管理コマンド処理の関係

Fig. 19 Schedule of top events and RBAC management commands.

の制限を受ける。したがって，システムの上流設計段階では，以下の条件を満足するように運用設計，システム設計を行うものとする。

- 故障木のトップ事象“TE1 セキュリティ違反”，“TE2 機会損失”は，理想時刻の前後に設ける許容範囲時間内はポリシー違反とはしない。
- 故障木のトップ事象“TE3 システム管理コスト”における開始時刻は，運用上の制限等で定められた開始可能時刻，完了時刻内で設定する。

これらの関係を，図 19 に示す。いずれも，TE1 または TE2 が許容範囲外にある場合を例示した。異種分散 RBAC リスク評価モデルを用いたシミュレーション結果を入力として，以下の手順で設計段階へのフィードバックを行う。

- モデル駆動評価により，RBAC モデルの管理コマンドに対応した中間事象，およびトップ事象が，図 19 に示す許容範囲内にあるか確認する。
- 許容範囲外にある場合は，リスク評価設計プロセスに戻り，以下の手順で異種分散 RBAC リスク評価モデル定義における実時間プロファイルを再設計する。
評価前のイベントの開始時刻を t_s ，終了時刻を t_e ，TE1，TE2 の許容範囲の開始時刻を p_s ，終了時刻を p_e ，とする。

(1-1)(2-1) 事前起動，許容範囲前完了 $t_e < p_s$

- イベントの開始時刻を指定するステレオタイプ RTArrivalPattern に $t_s + (p_s -$

$t_e) < t < t_s + (p_e - t_e)$ を満足する t を設定する．ただし，TE3 の制限内とする．

- プラットフォームの性能値 e を $(p_s - t_s)/(t_e - t_s) < e < (p_e - t_s)/(t_e - t_s)$ を満足するように低く設定する．

(1-2)(2-2) 事前起動，許容範囲後完了 $t_s < p_s \wedge p_e < t_e$

- イベントの開始時刻を指定するステレオタイプ RTArrivalPattern に $t < t_s - (t_e - p_e)$ を満足する t を設定する．ただし，TE3 の制限内とする．
- プラットフォームの性能値 e を $(p_s - t_s)/(t_e - t_s) < e < (p_e - t_s)/(t_e - t_s)$ を満足するように高く設定する．

(1-3)(2-3) 事後起動，許容範囲後完了 $p_s < t_s \wedge p_e < t_e$

- イベントを投入する時刻を指定するステレオタイプ RTArrivalPattern に $t < t_s - (t_e - p_e)$ を満足するように設定する．ただし，TE3 の制限内とする．もしくは運用を変更して，TE3 の制限を改善する．
- プラットフォームの性能値 e を $e < (p_e - t_s)/(t_e - t_s)$ となるように高く設定する．

- (3) 以上の設定を行い，再シミュレーションを行い，許容範囲内に収まることを確認するまで繰り返す．改善が見られない場合は，システム構成設計プロセスに戻り，ロール段数，プロビジョニング階層，動的ルール解釈の時点といったシステム構成の再設計，リスク評価再設計を行う．

5.3 世代管理 RBAC システムへ適用

4.3 節で示した世代管理 RBAC システムのシミュレーション結果をもとに考察する．

[仮定]

- TE1 セキュリティ違反，TE2 機会損失の許容範囲 ≤ 0.10
- 1 日あたりの運用管理コストの許容範囲 $\leq C_{\max}$
- 社員外ユーザを事後ランダム入力とし，機会損失が発生
- 社員外ユーザ投入バッチ処理に要する運用管理コスト $= C_1$

[シミュレーション結果①]

- 1 日 1 回のプロビジョニングとしてシステム構成設計，リスク設計を実施
- 過渡状態によって発生するセキュリティ違反の確率 $= 0.4844 > 0.10$
- 1 日あたりの運用管理コスト $= C_1$

以上の結果から，機会損失の許容範囲内の条件を満足できないことが判明した．この確率を改善するために，プロビジョニングのイベント投入スケジュールを示すステレオタイプ

RTArrivalPattern の周期を 1 日 1 回から 3 時間に 1 回として再シミュレーションを行う．

[シミュレーション結果②]

- 3 時間に 1 回のプロビジョニングとしてリスク設計を実施
- 過渡状態によって発生するセキュリティ違反の確率 $= 0.0653 < 0.10$
- 1 日あたりの運用管理コスト $= C_1 \times 8$

以上の結果，確率は，0.0653 となり条件を満足することが確認できる．このとき，運用管理コスト $C_1 \times 8 \leq C_{\max}$ であれば終了し， $C_1 \times 8 > C_{\max}$ であれば，再度，リスク設計，シミュレーションを実施する．

本評価では，社員外ユーザの投入に絞って実施した例を示したが，RBAC モデルのコマンド，システム関数，および異種 RBAC モデルを吸収する構成要素といったすべての事象がモデル，故障木で定義されるので，対象ユーザの増加，プロビジョニング先ターゲットシステムの増減といった局所的な変化に対する企業全体システムへの影響といった総合評価が可能である．

6. おわりに

本論文では，大規模企業に見られる組織構造，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リスク評価について示した．RBAC モデル適用には，セキュリティ，従業員の生産性，システム構築・運用コストを総合的に考慮する必要がある．しかし，それらは相反する要因であるため，トレードオフを考慮した総合的評価が必要とされる．さらに，システム設計段階でのリスク評価，および実システムの評価結果をフィードバックした設計手法が求められている．そこで本論文では，システム設計情報に処理負荷情報，入力情報を定義した実時間プロファイルを付加してシミュレーションを行い，その結果を，故障木を用いた RBAC モデル適用システムリスク評価方式に適用することによって設計段階で評価可能とした．また，世代管理 RBAC システムを題材に，提案した方式を用いた解析を行い，実システム性能測定データをもとに，シミュレーション結果を加えて，(a) RBAC 拡張モデル，およびプラットフォームの選択，(b) セキュリティ違反，機会損失の許容範囲内でのスケジュール設計に効果があることを示した．本論文で提案した分散システム処理システムのシミュレーションを利用した設計手法を利用することによって，端末等のエンドポイントへのパッチ情報，パターンファイルといったセキュリティ情報の配信や，エンドポイントのセキュリティ監視，ログ情報の収集の遅延によって発生するリスク評価に適用が可能である．今後は，こ

のようなアクセス制御以外のセキュリティ対策を加味した総合的リスク評価へ範囲を拡大していく所存である。

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, 2nd 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) OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2 (2007). <http://www.omg.org/spec/UML/2.1.2/>
- 11) OMG UML Profile for schedulability, Performance, and Time Specification Version 1.1 (2005). <http://www.omg.org/technology/documents/formal/schedulability.htm>
- 12) Douglass, B.P.: *Real Time UML Third Edition Advances in the UML for Real-Time Systems*, Addison-Wesley (2007).
- 13) OMG Systems Modeling Language (OMG SysML) Version 1.1 (2008). <http://www.omg.sysml.org/>
- 14) 近藤誠一, 白木宏明, 大沼聡久ほか: ロールベースアクセス制御情報の多バージョン並行処理制御を利用した監査ログトラッキング手法, 情報処理学会論文誌: データベース, Vol.46, No.SIG18 (TOD 28), pp.103-115 (2005).
- 15) Park, J., Costello, K., Neven, T., et al.: A Composite RBAC Approach for Large, Complex Organizations, *SACMAT'04* (2004).
- 16) Convington, M., Moyer, M. and Ahamad, M.: Generalized role-based access control for securing future applications, *Proc. National Information Systems Security Conference (NISSC)* (2000).
- 17) Moyer, M. and Ahamad, M.: Generalized Role-Based Access Control, *Proc. 2001 International Conference on Distributed Computing Systems (ICDCS)* (2001).
- 18) Gallaher, M., O'Connor, A. and Kropp, B.: The Economic Impact of Role-Based Access Control (NIST Planning Report 02-1) (2002).
- 19) Briney, A.: Security Focused, *Information Security* (2000).
- 20) Computer Security Institute: CSI Survey 2007, *The 12th Annual Computer Crime and Security Survey* (2007).
- 21) Basin, D., Doser, J. and Lodderstedt, T.: Model Driven security for Process-Oriented Systems, *SACMAT'03* (2003).
- 22) Ahn, G. and Hu, H.: Towards Realizing a Formal RBAC Model in Real Systems, *SACMAT'07* (2007).
- 23) Object Constraint Language, OMG Available Specification Version 2.0 (2006). <http://www.omg.org/spec/OCL/2.0>
- 24) Sohr, K., Drouineaud, M., Ahn, G., et al.: Analyzing and Managing Role-Based Access Control Policies, *IEEE Trans. Knowledge and Data Engineering*, Vol.20, No.7 (2008).
- 25) 木幡康博, 池田健一郎, 釜坂 等, 高橋洋一, 山足光義: 確実なセキュリティ運用を実現する統合 ID 管理システム “iDcenter”, 三菱電機技報 2009 年 9 月号, pp.35-38 (2009).
- 26) Kondo, S., Iwaihara, M., Yoshikawa, M. and Torato, M.: 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, Springer (2008).
- 27) 近藤誠一, 岩井原瑞穂, 吉川正俊, 虎渡昌史: 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, 情報処理学会論文誌, Vol.50, No.11, pp.2727-2739 (2009).

(平成 22 年 9 月 27 日受付)

(平成 23 年 2 月 4 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，ウォータールー大学客員准教授。XML 情報検索，異種情報の統合利用等の研究に従事。電子情報通信学会，ACM 各会員。



小宮 崇

2001 年佐賀大学大学院工学系研究科電子工学専攻修了。同年三菱電機（株）入社。情報セキュリティシステムに関する研究開発に従事。



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 年より三菱電機インフォメーションシステムズ（株）に転籍。博士（情報学）。情報セキュリティシステムに関する研究開発に従事。