情報処理学会論文誌　Vol. 51　No. 3　728–739 (Mar. 2010)

分散システムにおけるロールベースアクセス制御の更新問題

朝倉義晴[†1,†2]　中本幸一[†2]

ロールベースアクセス制御（RBAC）では，システム内の情報にアクセスするための特権をロールに関連付け，アクセス者にロールを割り当てることで，アクセスを制御する．分散システム内の情報を RBAC で保護する場合，システム管理者が各コンピュータ上にロールを定義し，コンピュータ間のロールを適切にマッピングする必要がある．このとき，運用環境の変化によりロール定義を更新すると，ロールのマッピングが適切に維持されるかを確認しなければならない．この確認は，管理者にとって大きな負荷となる．本論文では，ロール定義の更新にともなう管理者の負荷を軽減するために，ロール間の関係を定義し，その関係を維持するロール定義の更新操作を提案する．提案する更新操作を用いることで，ロールのマッピングが適切に維持されるかを確認することなくロール定義の更新が可能となり，ロール定義の更新にともなう管理者の負荷を軽減することができる．

Renewal Problems in Role-based Access Control for a Distributed System

Yoshiharu Asakura[†1,†2] and Yukikazu Nakamoto[†2]

In Role-based access control (RBAC), access control to information in a computer system is done by associating access privileges with a role and assigning the role to a subejct. When system administrators protect information in a distributed system by applying RBAC, they need to define roles on each computer and map roles between computers appropriately. If they renew role definitions because of changes of operating environment, they have to check whether mapping of roles is preserved appropriately. This check is a heavy work load for them. In this paper, in order to reduce their work loads, we define a relationship between roles and propose transformation manipulations that preserve the relationship. By using the transformation manipulations, they can renew role definitions without checking whether mapping of roles is preserved appropriately. Therefore, we can reduce their work loads.

1. はじめに

サーバ，パソコン，家電機器などの複数のコンピュータがネットワークを介して接続され，それらのコンピュータの協調動作により 1 つのシステムを実現する分散システムが期待されている．分散システムの協調動作では，コンピュータ（アクセス元コンピュータ）による異なるコンピュータ（アクセス先コンピュータ）上のファイルなどのオブジェクトへのアクセスやサービスの利用が必要となる場合がある．このとき，コンピュータが自由に機密情報や個人情報を含むオブジェクトにアクセスできたり，サービスを利用できたりするのは好ましくない．オブジェクトへのアクセスを制限するために，サーバシステムではアクセス制御が利用されている[1)–3)]．アクセス制御は，サブジェクト（たとえば，プロセス）にアクセスのための適切な特権を認可することで，サブジェクトによるオブジェクト（たとえば，ファイル）への操作（たとえば，読み込み）を制御する．前述の要求を実現するためには，サブジェクトに適切な特権を認可する必要があるが，多数存在するサブジェクトへの個別の特権の認可は非常に繁雑な作業であり一般的にコストが大きい．認可に要するコストを削減するためのアクセス制御モデルとして，ロールベースアクセス制御[4),5)]（Role-based access control，以下 RBAC と略す）が提案されている．RBAC では，ロールを単位としてサブジェクトに特権を認可する．そのため，企業内の業務ごとの特権を関連付けたロールをあらかじめ定義しておき，業務実施者（サブジェクト）にそのロールを割り当てることで，サブジェクトに適切な特権を容易に認可することができる．また，実施する業務の変更にともなう認可する特権の変更もロールの割当てを変更するだけでよく，管理が容易となる．

分散システムにおいて RBAC に基づくアクセス制御を行う場合，分散システムに参加する各コンピュータにおいて，コンピュータの管理者がロールを定義する．コンピュータ内のオブジェクトにアクセスしたりサービスを利用したりするサブジェクトは，そのコンピュータ内のサブジェクト（ローカルサブジェクト）や異なるコンピュータ上のサブジェクト（リモートサブジェクト）である．これらローカルサブジェクトやリモートサブジェクトにロールを割り当てることで，アクセス制御を実現する．各コンピュータ上のロールはそのコン

†1 NEC システムプラットフォーム研究所
System Platforms Research Laboratories, NEC Corporation
†2 兵庫県立大学大学院応用情報科学研究科
Graduate School of Applied Informatics, University of Hyogo

ピュータの管理者により定義されるため，同一のロールが定義されているとは限らない．したがって，リモートサブジェクトに割り当てるロールは，アクセス元コンピュータ上でリモートサブジェクトに割り当てられているロールに基づいて適切に決める必要がある．このリモートサブジェクトへのロールの適切な割当ては，アクセス元コンピュータとアクセス先コンピュータ間のロールの適切なマッピングと見なせる．またコンピュータ上のロール定義は静的なものではなく，分散システムの運用中に更新されうる．アクセス先コンピュータ上のロール定義を更新した場合，定義を更新したロールに関連するマッピングが適切に維持されるかを確認しなければならない．アクセス元コンピュータの数が多い場合，ロールに関連するマッピングの数も多くなる．そのため，この確認，変更作業が膨大となり，コンピュータの管理者の負荷が高くなる．

本論文では，ロール定義の更新にともなう管理者の負荷を軽減するために，ロール間の関係を定義し，その関係を維持するロール定義の更新操作を提案する．提案する更新操作を用いることで，ロールのマッピングが適切に維持されるかを確認することなくロール定義の更新が可能となり，ロール定義の更新にともなう管理者の負荷を軽減することができる．

以後の本論文の構成は以下のとおりである．最初に 2 章で本論文の解決する問題を定義する．3 章では，本論文が基づく研究の諸定義について述べる．4 章で，ロール間の関係である拡張関係を定義し，拡張関係を維持しながらロール定義の更新を可能にする拡張変換操作を定義する．5 章で 4 章の定義を用いてロール定義の更新について議論し，6 章でプロジェクト内のファイルサーバを題材とするロール定義の更新事例を述べる．最後に 7 章で本論文の結論と今後の課題を述べる．

2. 問題定義

本章では，本論文で解決する問題を定義する．問題を定義する前に，分散システムに RBAC を適用する既存の研究について述べる．そして既存の研究をふまえ，問題定義を行う．

2.1 関連研究

分散システムに属するコンピュータ間のロールのマッピングに関して，いくつかの研究がなされている．文献 6) では，publish-subscribe モデルに基づく分散システムに RBAC を適用する distributed role based access control model（DRBAC）を提案している．DRBAC では，リソース提供者が分散ロールを定義する．この分散ロールはリソース利用者に提供され，リソース利用者が自身のロールを分散ロールにマッピングすることで，コンピュータ間のロールをマッピングする．文献 7) では，イントラネット下の分散システムに RBAC を適用する I-RBAC を提案している．I-RBAC では，イントラネットに属する全コンピュータに共通するグローバルロールと各コンピュータ独自のローカルロールが定義される．グローバルロールはローカルロールの集合とマッピングされ，グローバルロールが割り当てられたサブジェクトには，各コンピュータにおいてグローバルロールにマッピングされたローカルロールが割り当てられる．文献 8) では，コンピュータ間のロールのマッピングではなく，デリゲーション（delegation）によりロールの割当てを解決する dRBAC を提案している．dRBAC では，サブジェクトにロールを割当て可能かどうかを，分散システムに属するコンピュータ上に分散定義されているデリゲーションの遷移により判定する．デリゲーションを用いる場合でも，ロール定義の更新にともないデリゲーションが適切であるかを確認する必要がある．

2.2 本論文で解決する問題

1 章で述べたように，分散システムにおいて RBAC に基づくアクセス制御を実現するためには，コンピュータ間のロールを適切にマッピングする必要がある．ロールのマッピングの例として，以下に述べるプロジェクト A を考える．プロジェクト A の分散システムは，プロジェクトファイルを格納するファイルサーバ（アクセス先コンピュータ）とファイルサーバにアクセスする複数のクライアント（アクセス元コンピュータ）から構成される．プロジェクト A の参加者は，プログラマ，営業担当者，システム管理者，ファイルサーバ管理者である．プログラマと営業担当者は複数の部門に所属しており，部門内のクライアントを利用してプロジェクト A を遂行する．システム管理者は部門ごとにいて，部門内のクライアントを管理する．ファイルサーバ管理者は，ファイルサーバを管理する．プロジェクト内において適切なアクセス制御を実現するために，システム管理者は自身の管理するクライアント内にロール（たとえば，プログラマ用の *LProgrammer*）を定義し，ローカルサブジェクト（たとえば，プログラマ）にロールを割り当てる．またファイルサーバ管理者は，ファイルサーバ内にロール（たとえば，プログラマ用の *SProgrammer*）を定義し，リモートサブジェクト（たとえば，クライアントを利用しているプログラマ）にロールを割り当てる．*LProgrammer* にはソースファイルの読み書き特権とコンパイラの使用特権が関連付けられており，*SProgrammer* にはソースファイルの読み書き特権，コンパイラと検査ツールの使用特権が関連付けられている．プログラマにこれらのロールを割り当てることで，プログラマはソースファイルをコンパイルし，ファイルサーバにおいて検査ツールを利用できる．このようにプログラマがプロジェクトを遂行するためには，プログラマに適切なロールを割り当てる必要があり，*LProgrammer* を *SProgrammer* にマッピングする必

要がある．

また 1 章で述べたように，プロジェクト遂行中にロール定義を更新する場合も考えられる．たとえば最小特権の観点から，各ロールに割り当てる特権を細分化することでプロジェクト参加者に割り当てる特権の範囲をきめ細かく制御することを考える．この考えに基づき，コンパイラの使用特権と検査ツールの使用特権を異なるロールに割り当てるように，ファイルサーバ上のロール定義を更新する．この更新は，検査ツールの使用特権を関連付けたロールをファイルサーバ上に新たに定義し，検査ツールの使用特権の関連付けを $SProgrammer$ から削除することで実現できる．しかしこの場合，プログラマが検査ツールを使用できなくなる．プログラマが引き続き検査ツールを使用できるようにするためには，$LProgrammer$ と検査ツールの使用特権を関連付けたロールとの新たなマッピングが必要となる．この例のように，ロール定義の更新によりロールの適切なマッピングが維持されない場合もあるため，ロール定義の更新の際にはロールの適切なマッピングが維持されるかを確認しなければならない．ロール定義の異なるクライアントの数が多い場合，マッピング元のロールの種類が多くなるため，確認すべきロールの適切なマッピングの数も多くなる．そのため，ロール定義の更新にともなうサーバ管理者の負荷が高くなるという問題がある．

2.1 節で述べた研究では，コンピュータ間のロールをマッピングするためのモデルについて述べている．しかしこれらの研究では，分散システムの運用中にロール定義を更新することを考慮していない．すなわち，アクセス先コンピュータのロール定義の更新後，ロールの適切なマッピングが維持されることを保証できない．なぜなら既存の研究では，以下の 2 つの問題を考慮していないためである．

P1：コンピュータ間のロールのマッピングに関して，ロールの適切なマッピングの維持を定義していない．

P2：アクセス先コンピュータのロール定義の更新に関して，ロールの適切なマッピングを維持するためのロール定義の更新操作を定義していない．

これまでは **P1** のため，ロールの適切なマッピングの維持が議論されることはなかった．また **P1** が未定義なため，**P2** が定義されることもなかった．これら 2 つの問題のため，アクセス先コンピュータのロール定義を更新するごとに，ロールのマッピングが適切であるかを確認する必要があった．本論文では，ロール間の関係とロール定義の更新操作を形式的に定義することによりこれら 2 つの問題を解決することで，ロールの適切なマッピングを維持しながらロール定義の更新を可能とする．すなわち，ロール定義の更新時にロールの適切なマッピングの確認作業が不要となり，アクセス先コンピュータの管理者の負荷を軽減できる．

3. 諸 定 義

筆者らは，管理者によるロールの効率的な設定を支援するために，ロールグラフ[9),10)] を拡張した拡張ロールグラフを提案し，拡張ロールグラフ間の等価関係，および，等価変換操作を定義した[11)]．本論文では，これらの研究に基づき 2.2 節の問題を解決する．本章では，これらの研究の諸定義を説明する．

3.1 ロールグラフと拡張ロールグラフ

ロールグラフ[9),10)] は，ノードがロールを，辺がロールの継承関係を表す有向非循環グラフで，ロール集合を表す RS と辺集合を表す ES の組である (RS, ES) で表される．ロールグラフの例を**図 1** に示す．ロール r_2 がロール r_1 を継承している場合，r_2 は r_1 に関連付けられているすべての特権を継承し（すなわち，r_1 に関連付けられているすべての特権は r_2 にも関連付けられ），r_1 は r_2 のジュニアロール（junior role）と呼ばれる．また，r_2 は r_1 のシニアロール（senior role）と呼ばれる．ロールグラフ中に辺 (r_1, r_2) が存在する場合，r_1 は r_2 の直ジュニアロール（immediate junior role）と呼び，r_2 は r_1 の直シニアロール（immediate senior role）と呼ぶ．図 1 を例にすると，r_1 は r_2，r_3，r_4 のジュニアロールであり，r_4 は r_1，r_2 のシニアロールである．また，r_1 は r_2，r_3 の直ジュニアロールであり，r_2，r_3 は r_1 の直シニアロールである．

ロール r に関連付けられる特権は，（特権 1）r に直接割り当てられる特権，（特権 2）r のジュニアロールから継承する特権，（特権 3）r に直接割り当てられ，かつ，r のジュニアロールから継承する特権，の 3 つに分類される．本論文では文献 10) と同様に，特権 1 を r のダイレクト特権（direct privilege）として，特権 1 と特権 2 を r のイフェクティブ特

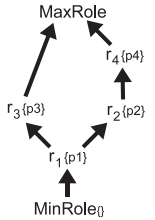

図 1　ロールグラフの例
Fig. 1　Sample role graph.

権（effective privilege）[⋆1] として定義している．また，ロール r のダイレクト特権集合を $Direct(r)$ と記述し，イフェクティブ特権集合を $r.rpset$ と記述する．特権 3 は r のジュニアロールを jr とすると，$p \in Direct(r) \wedge p \in Direct(jr)$ を満たす特権 p であり，本論文では余剰特権と呼ぶ．図 1 では，{，} の中に各ロールのダイレクト特権を記述している．図 1 の r_2，r_4 のイフェクティブ特権集合は，$r_2.rpset = \{p1, p2\}$，$r_4.rpset = \{p1, p2, p4\}$ である．

ロールグラフは，ロールグラフ特性（Role Graph Properties）[10] と呼ばれる次の特性を持つ．

(1)　$MaxRole$ を 1 つ持つ．
(2)　$MinRole$ を 1 つ持つ．
(3)　ロールグラフは非循環グラフである．
(4)　$MinRole$ から任意のロールへのパス[⋆2]が存在する．
(5)　任意のロールから $MaxRole$ へのパスが存在する．
(6)　任意の 2 つのロール r_i，r_j に対して，$r_i.rpset \subset r_j.rpset$ ならば，r_i から r_j へのパスが存在しなければならない．

ここで $MaxRole$ はロールグラフ中のすべてのロールの特権を関連付けたロールであり，$MinRole$ はロールグラフ中のすべてのロールに共通する特権（共通する特権を持たない場合は，$MinRole.rpset = \emptyset$ となる）を関連付けたロールである．ロールグラフは，well-formedness [9] と呼ばれる特性も持つ．well-formed なロールグラフは，余剰辺と余剰特権を持たない．ここで余剰辺とは，ロール r_i と r_j 間に辺 (r_i, r_j) とこれ以外のパスが 1 つ以上存在するときの (r_i, r_j) である．これら 2 つの特性より，ロールグラフは有向非循環グラフで推移簡約である．よって，ロールグラフの形状はロール集合に対して一意に定まる [12]．

筆者らは，ロールの効率的な定義を実現するために，ロールグラフを拡張した拡張ロールグラフを提案した [11]．拡張ロールグラフは，ロールグラフが持つ 2 つの特性（ロールグラフ特性と well-formedness）を緩和している．そのため，拡張ロールグラフの形状はロール集合に対して一意に定まらず，多様な形状をとりうる．よって管理者は，拡張ロールグラフを管理者の望む形状に柔軟に変換できる．拡張ロールグラフは，拡張ロール集合を表す ERS と辺集合を表す ES の組である (ERS, ES) で表される．拡張ロール集合は，抽象ロールを含むように拡張されたロール集合である．ここで，サブジェクトを割当てできないロールを抽象ロールと定義する．拡張ロールグラフは，以下のすべての条件を満たすロールグラフである．

C1：ロールグラフ特性の特性 1 から 5 をすべて満たす．
C2：0 個以上の抽象ロールを含む．
C3：0 個以上の余剰辺と 0 個以上の余剰特権を持つ．

この定義においてロールグラフは，拡張ロールグラフの抽象ロール，余剰辺，余剰特権が 0 個で，かつ，拡張ロールグラフがロールグラフ特性の特性 6 を満たす特別な場合と見なせる．

3.2　等 価 関 係

拡張ロールグラフの等価関係 [11] は，下記に示す一連の定義により定義される．

定義 1（ロールの等価性）ロール r_i と r_j に対して，r_i と r_j が等価（$r_i = r_j$ と記述する）となる必要十分条件は，$r_i.rpset = r_j.rpset$ である．

定義 2（ロール集合の包含関係）ロール集合 RS_i と RS_j に対して，RS_i が RS_j に包含される（$RS_i \subseteq RS_j$ と記述する）ための必要十分条件は，任意のロール $r_i \in RS_i$ に対して，$r_i = r_j$ となるようなロール $r_j \in RS_j$ が存在することである．

定義 3（拡張ロール集合の等価性）拡張ロール集合 ERS_1 と ERS_2 に対して，ERS_1 が ERS_2 と等価（$ERS_1 = ERS_2$ と記述する）となる必要十分条件は，$(ERS_1 - VRS_1) \subseteq (ERS_2 - VRS_2) \wedge (ERS_1 - VRS_1) \supseteq (ERS_2 - VRS_2)$ である．ここで VRS_1 と VRS_2 は，それぞれ ERS_1 と ERS_2 に含まれる抽象ロール集合を表す．

定義 4（拡張ロールグラフの等価性）拡張ロールグラフ $ERG_1 = (ERS_1, ES_1)$ と $ERG_2 = (ERS_2, ES_2)$ に対して，ERG_1 が ERG_2 と等価（$ERG_1 = ERG_2$ と記述する）となる必要十分条件は，$ERS_1 = ERS_2$ である．

ロールグラフは拡張ロールグラフの特別な場合と見なせるため，定義 3，4 はロールグラフに対しても成り立つ．

拡張ロールグラフの変換操作は，変換操作の適用前後において定義 4 で述べた等価関係を満たす変換操作と満たさない変換操作の 2 つに分類でき，前者の変換操作を等価変換操作 [11] と呼ぶ．等価変換操作を用いることで，等価関係を維持しながら拡張ロールグラフを変換することが可能となる．以下に 5 つの等価変換操作を述べ，変換操作例を**図 2** に示す．

特権の分配（PD）：ダイレクト特権をすべての直シニアロールに分配する変換操作である．抽象ロール vr に対し，あるダイレクト特権 $cp \in Direct(vr)$ が存在するとき，cp を $Direct(vr)$ から削除し，vr のすべての直シニアロールのダイレクト特権集合に cp を

⋆1 すなわち，特権 p がロール r のダイレクト特権であれば，p は r のイフェクティブ特権でもある．
⋆2 ロール r_1 からロール r_2 へ 1 つ以上の辺を経由して到達可能なとき，r_1 から r_2 へのパスが存在するという．

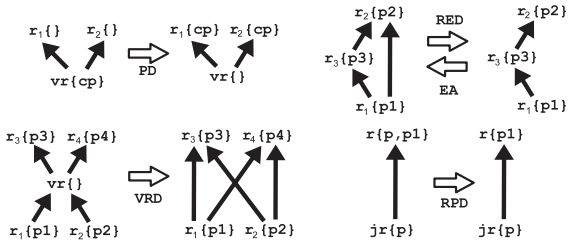

図 2　等価変換操作の例
Fig. 2　Example of equivalent transformation manipulations.

追加する．

抽象ロールの削除（VRD）： 拡張ロールグラフから抽象ロールを削除する変換操作である．抽象ロール vr に対し，$Direct(vr) = \emptyset$ を満たすとき，vr を拡張ロールグラフから削除する．このとき，vr のすべての入射辺と出射辺を削除し，vr のすべての直ジュニアロールから vr のすべての直シニアロールへの辺を追加する．

辺追加（EA）： 拡張ロールグラフに辺を追加する変換操作である．ロール r_1 と r_2 に対し，$r_1.rpset \subseteq r_2.rpset$ を満たすとき，辺 (r_1, r_2) を追加する．

余剰辺削除（RED）： 拡張ロールグラフから余剰辺を削除する変換操作である．ロール r_1 と r_2 に対し，辺 (r_1, r_2) が余剰辺であるとき，(r_1, r_2) を削除する．

余剰特権削除（RPD）： ロールから余剰特権を削除する変換操作である．ロール r のあるダイレクト特権 $p \in Direct(r)$ に対し，$p \in Direct(jr)$ を満たす r のジュニアロール jr が存在するとき，p を $Direct(r)$ から削除する．

等価関係を満たさない変換操作は，拡張ロールグラフのアクセス権限を拡大する変換操作と縮小する変換操作とにさらに分類される．拡大する変換操作としては，拡張ロールグラフへのロールの追加，ロールへの新しいダイレクト特権の追加，などがあげられる．縮小する変換操作としては，拡張ロールグラフからの無条件なロールの削除，ロールからの無条件なダイレクト特権の削除，などがあげられる．

4. 拡 張 関 係

コンピュータのロール定義を更新するとき，拡張ロールグラフを用いてロール定義を更新すると効率が良い．よって，拡張ロールグラフを変換することでロール定義を更新する．2.2 節で述べた問題を解決するためには，等価関係のような拡張ロールグラフ間の関係が必要である．しかし 3.2 節で述べた等価変換操作では，拡張ロールグラフに新しいロールや特権を追加することはできない．すなわち，コンピュータに新しいアクセス権限を追加することができない．そのため，新しいロールや特権の追加を考慮した拡張ロールグラフ間の新しい関係，および，変換操作が必要とされる．本章では拡張ロールグラフ間の新しい関係として拡張関係を定義し，拡張ロールグラフの変換時に拡張関係を維持することを保証する変換操作である拡張変換操作[⋆1] を定義する．

4.1　拡張ロールグラフの拡張関係

本節では，ロール間，拡張ロール集合間，拡張ロールグラフ間の関係の 1 つである拡張関係を定義する．拡張関係を定義する前に，拡張ロール集合を再定義する．従来の拡張ロール集合[11] は，等価なロールを 2 つ以上含むことができなかった．2 つ以上の等価なロールの存在を許容することで，拡張関係の議論を円滑にできる．

定義 5（拡張ロール集合の再定義） 拡張ロール集合とは，以下のいずれかの条件を満たすロール集合である．

- 抽象ロールを少なくとも 1 つ含む．
- 等価なロールの組を少なくとも 1 つ含む．

最初に 2 つのロール間の拡張関係を定義する．ロール間の拡張関係は拡張関係の基礎となる定義であり，各ロールのイフェクティブ特権集合の包含関係を用いて定義される．次にロール間の拡張関係の定義を，拡張ロール集合間の拡張関係に拡大する．最後に，拡張ロール集合間の拡張関係を用いて拡張ロールグラフ間の拡張関係を定義する．

定義 6（ロール間の拡張関係） 2 つのロール r_1，r_2 において，$r_1.rpset \subset r_2.rpset$ が成り立つとき，r_2 は r_1 に対して拡張関係にあるといい，$r_1 < r_2$ と記述する．

拡張ロール集合間の拡張関係を定義する前に，あるロールと等価関係，もしくは，拡張関係にあるロール集合を返す関数を定義する．

定義 7（関数） 拡張ロール集合 ERS_1，ERS_2 において，ERS_1 に属する任意のロール r_1（ただし，$r_1 \neq MinRole \wedge r_1 \neq MaxRole$）に対して $r_1 \leq r_2$ を満たすすべての $r_2 \in ERS_2$（ただし，$r_2 \neq MinRole \wedge r_2 \neq MaxRole$）からなる集合を返す関数を $\Gamma_{ERS_1 \to ERS_2}(r_1)$ と記述する．また，ERS_2 に属する任意のロール r_2（ただし，$r_2 \neq MinRole \wedge r_2 \neq MaxRole$）に対して $r_1 \leq r_2$ を満たすすべての $r_1 \in ERS_1$（ただし，$r_1 \neq MinRole \wedge r_1 \neq MaxRole$）からなる集合を返す関数を $\Gamma^{-1}_{ERS_1 \to ERS_2}(r_2)$ と記述する．

⋆1 拡張変換操作は，3.2 節で述べた拡張ロールグラフのアクセス権限を拡大する変換操作である．

定義 7 の関数を用いて，拡張ロール集合間の拡張関係は定義される．

定義 8（ロール集合間の拡張関係） 等価でない拡張ロール集合 ERS_1，ERS_2 において，ERS_2 が ERS_1 に対して拡張関係（$ERS_1 < ERS_2$ と記述する）にあるための必要十分条件は，ERS_1 に属する任意のロール r（ただし，$r \neq MinRole \wedge r \neq MaxRole$）と拡張関係，または，等価関係にあるロールが $MinRole$ と $MaxRole$ を除いた ERS_2 に存在することである．形式的に記述すると，$\forall r \in (ERS_1 - \{MaxRole, MinRole\}), \Gamma_{ERS_1 \to ERS_2}(r) \neq \emptyset \Leftrightarrow ERS_1 < ERS_2$ である．

拡張ロールグラフ間の拡張関係は，以下のように定義される．

定義 9（ロールグラフ間の拡張関係） 拡張ロールグラフ $ERG_1 = (ERS_1, ES_1)$，$ERG_2 = (ERS_2, ES_2)$ において，ERG_2 が ERG_1 に対して拡張関係（$ERG_1 < ERG_2$ と記述する）にあるための必要十分条件は，$ERS_1 < ERS_2$ である．

2 つの拡張ロールグラフ $ERG_1 = (ERS_1, ES_1)$ と $ERG_2 = (ERS_2, ES_2)$ に対して $ERG_1 < ERG_2$ が成り立つとき，ERS_1 に含まれる任意のロールと等価関係，もしくは，拡張関係にあるロールが ERS_2 に存在する．なお，定義 7，8，9 は拡張ロール集合間を対象とした定義であるが，ロール集合間，あるいは，ロール集合と拡張ロール集合間でも成り立つ．

次に，以後の議論を容易にするために，補題を 4 つ示す．任意の 2 つの拡張ロール集合 ERS_1，ERS_2 に対して，以下の補題が成り立つ．

補題 1　$ERS_1 \leq ERS_2$，$r_2 \in ERS_2$ に対して，$\Gamma^{-1}_{ERS_1 \to ERS_2}(r_2) = \emptyset$ の場合，$ERS_1 \leq (ERS_2 - \{r_2\})$ が成り立つ．

証明　$ERS_1 \leq (ERS_2 - \{r_2\})$ が成り立たないと仮定する．今，$ERS_1 \leq ERS_2$ は成り立つため，仮定より $\Gamma_{ERS_1 \to ERS_2}(r_1) = \{r_2\}$ を満たすロール $r_1 \in ERS_1$ が少なくとも 1 つ存在する．これは，$\Gamma^{-1}_{ERS_1 \to ERS_2}(r_2) = \emptyset$ と矛盾する．したがって，$ERS_1 \leq (ERS_2 - \{r_2\})$ が成り立つ．□

補題 2　任意のロール r に対して，$ERS_1 \leq ERS_2 \Rightarrow ERS_1 \leq (ERS_2 \cup \{r\})$ が成り立つ．

証明　ロール r と等価なロールが ERS_2 に属する場合と属さない場合とに分ける．

- ロール r と等価なロールが ERS_2 に属する場合
  $ERS_2 = ERS_2 \cup \{r\}$ が成り立つ．$ERS_1 \leq ERS_2$ が成り立つため，$ERS_1 \leq (ERS_2 \cup \{r\})$ が成り立つ．
- ロール r と等価なロールが ERS_2 に属さない場合
  ERS_2 にロール r を追加して ERS_3（すなわち $ERS_3 = ERS_2 \cup \{r\}$）になったと仮定する．今，ERS_2 に属する任意のロール r' に対して $r' \in \Gamma_{ERS_2 \to ERS_3}(r')$ が成り立つため，定義 8 より $ERS_2 < ERS_3$ が成り立つ．$ERS_1 \leq ERS_2$ が成り立つため，$ERS_1 < ERS_3$ が成り立つ．

以上より，任意のロール r に対して，$ERS_1 \leq ERS_2 \Rightarrow ERS_1 \leq (ERS_2 \cup \{r\})$ が成り立つ．□

補題 3　$r_1 < r_2 \wedge p \notin r_1.rpset \wedge p \in r_2.rpset$ を満たす特権 p に対して，イフェクティブ特権集合が $r_2.rpset - \{p\}$ であるロールを r_3（すなわち $r_3.rpset = r_2.rpset - \{p\}$）とすると，$r_1 \leq r_3$ が成り立つ．

証明　$r_1 < r_2$ と定義 6 より，$r_1.rpset \subset r_2.rpset$ が成り立つ．今，$p \notin r_1.rpset$ が成り立つため，$r_1.rpset \subseteq r_2.rpset - \{p\}$ が成り立つ．よって，$r_3.rpset = r_2.rpset - \{p\}$ より，$r_1.rpset \subseteq r_3.rpset$ が成り立つ．したがって定義 6 より，$r_1 \leq r_3$ が成り立つ．□

補題 4　拡張ロール集合 ERS に属するロール r_1, r_2 が $Direct(r_1) = \emptyset \wedge r_1.rpset = r_2.rpset$ を満たすとき，$ERS = ERS - \{r_1\}$ が成り立つ．

証明　r_1 のダイレクト特権集合が空集合であるため，r_1 を削除しても r_1 以外の ERS に属するロールのイフェクティブ特権集合には影響を及ぼさない．また，r_1 と等価なロール r_2 が存在するため，ロール集合の等価性の定義より，$ERS = ERS - \{r_1\}$ が成り立つ．□

4.2　拡張変換操作

本節では，拡張変換操作について述べる．拡張変換操作とは，拡張ロールグラフ $ERG_1 = (ERS_1, ES_1)$ に変換操作を N 回適用して拡張ロールグラフ $ERG_{N+1} = (ERS_{N+1}, ES_{N+1})$ を得たときに，$ERG_1 \leq ERG_{N+1}$ を満たすことを保証する変換操作である．なお，ERG_1 から ERG_{N+1} への変換過程において，等価変換操作を用いてもかまわない．等価変換操作集合を $\mu_=$，拡張変換操作集合を $\mu_<$ とすると，ERG_1 から ERG_{N+1} への変換は，$ERG_1 \overset{\tau_1}{\to} ERG_2 \overset{\tau_2}{\to} \cdots \overset{\tau_N}{\to} ERG_{N+1}$, $\tau_k \in (\mu_= | \mu_<)$，$1 \leq k \leq N$ という過程で表される．ここで，変換操作を適用する元となった拡張ロールグラフ ERG_1 を基点ロールグラフと呼ぶ．また任意の $r \in (ERS_1 - \{MinRole, MaxRole\})$ に対して，$\Gamma_{ERS_1 \to ERS_k}(r) \neq \emptyset$，$2 \leq k \leq N+1$ が成り立つ．

拡張ロールグラフの変換過程において，2 つ以上の等価なロールが拡張ロールグラフに存在する場合がある．このような拡張ロールグラフを非正規拡張ロールグラフと呼ぶ．非正規拡張ロールグラフは，拡張ロールグラフの変換過程において一時的に現れる場合にのみ許容される．拡張ロールグラフの変換の結果，最終的に得る拡張ロールグラフは非正規拡張ロールグラフであってはならない．非正規拡張ロールグラフの定義を定義 10 に述べる．

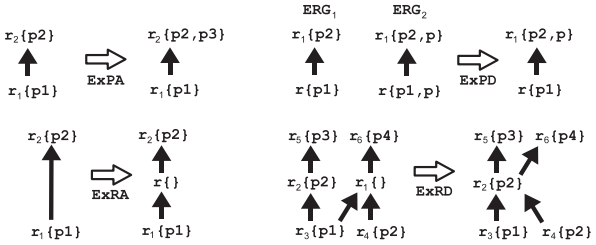

図 3　変換操作の例
Fig. 3　Example of transformation manipulations.

定義 10（非正規拡張ロールグラフ） 拡張ロールグラフ $ERG = (ERS, ES)$ において，$\exists r_1, r_2 \in (ERS - VRS - \{MaxRole, MinRole\})$，$r_1.rpset = r_2.rpset$ が成り立つ場合，ERG を非正規拡張ロールグラフと呼ぶ．ここで VRS は ERG に存在する抽象ロール集合で，$VRS \subset ERS$ である．

最初に変換操作として，以下の 4 つの操作を述べ，変換操作例を**図 3** に示す．以下の操作の説明では，基点ロールグラフを $ERG_1 = (ERS_1, ES_1)$ とし，ERG_1 と拡張関係にある拡張ロールグラフ $ERG_2 = (ERS_2, ES_2)$ に対して変換操作を適用する．

特権の追加（ExPA）： ロールにダイレクト特権を追加する操作である．任意のロールに対して，ダイレクト特権を追加する．

特権の削除（ExPD）： ロールからダイレクト特権を削除する操作である．ロール $r \in ERS_2$ とダイレクト特権 $p \in Direct(r)$ に対して，直シニアロール集合 $ISeniors(r)$ に属する $MaxRole$ を除くすべてのロールがダイレクト特権 p を持ち，かつ，逆関数 $\Gamma^{-1}_{ERS_1 \to ERS_2}(r)$ に属するすべてのロールのイフェクティブ特権集合が p を含まないとき，p を $Direct(r)$ から削除する．削除できる条件を形式的に記述すると，$\forall r_1 \in (ISeniors(r) - \{Maxrole\})$，$\forall r_2 \in \Gamma^{-1}_{ERS_1 \to ERS_2}(r)$，$p \in Direct(r_1) \wedge p \notin r_2.rpset$ である．

ロールの追加（ExRA）： ロールグラフにダイレクト特権を持たないロールを追加する操作である．ロール r_1，$r_2 \in ERS_2$ に対して $r_1.rpset \subseteq r_2.rpset$ が成り立つとき，入射辺が (r_1, r) で出射辺が (r, r_2) であるロール r を ERS_2 に追加する．このとき辺 (r_1, r_2) が存在する場合，(r_1, r_2) を削除する．

ロールの削除（ExRD）： ロールグラフからロールを削除する操作である．ロール $r_1, r_2 \in ERS_2$ に対して $r_1 \notin ERS_1 \wedge Direct(r_1) = \emptyset \wedge r_1.rpset = r_2.rpset$ が成り立つとき，r_1 を ERS_2 から削除する．r_1 へのすべての入射辺は r_2 へ入射するように辺を付け替え，r_1 からのすべての出射辺は r_2 から出射するように辺を付け替える．

次に，上記 4 つの変換操作が拡張変換操作であることを定理 1 で示す．

定理 1（拡張変換操作） 変換操作 ExPA，ExPD，ExRA，ExRD は拡張変換操作である．

証明　拡張ロールグラフ $ERG_1 = (ERS_1, ES_1)$ と $ERG_2 = (ERS_2, ES_2)$ が等価関係，または，拡張関係（すなわち $ERG_1 \leq ERG_2$）にあり，ERG_1 が基点ロールグラフであるとする．このとき，ERG_2 に ExPA，ExPD，ExRA，ExRD を適用して得られる拡張ロールグラフ $ERG_3 = (ERS_3, ES_3)$ が $ERG_1 \leq ERG_3$ を満たすことを示す．

ExPA： ロール $r_2 \in ERS_2$ にダイレクト特権 p を追加してロール $r_3 \in ERS_3$ になったと仮定する．さらに r_2 の任意のシニアロール $sr_2 \in ERS_2$ のイフェクティブ特権集合に p を追加した r_3 のシニアロールを $sr_3 \in ERS_3$ と仮定する．このとき，定義 6 より $r_2 < r_3$ が成り立つ．また，sr_3 のイフェクティブ特権集合は sr_2 のイフェクティブ特権集合と等価か特権 p が増えているため，$sr_2 \leq sr_3$ が成り立つ．よって，$\Gamma_{ERS_2 \to ERS_3}(r_2) \neq \emptyset \wedge \Gamma_{ERS_2 \to ERS_3}(sr_2) \neq \emptyset$ が成り立つ．また，r_2 と任意の sr_2 以外のロール $r' \in ERS_2$ に対して，$r' \in \Gamma_{ERS_2 \to ERS_3}(r')$ であるため，$\Gamma_{ERS_2 \to ERS_3}(r') \neq \emptyset$ が成り立つ．よって，任意のロール $r \in ERS_2$ に対して $\Gamma_{ERS_2 \to ERS_3}(r) \neq \emptyset$ が成り立つ．ここで $ERS_2 \neq ERS_3$ のとき，定義 8 より $ERS_2 < ERS_3$ が成り立つ．$ERS_2 = ERS_3$ の場合[⋆1]も考慮すると，$ERS_2 \leq ERS_3$ が成り立つ．仮定より $ERS_1 \leq ERS_2$ が成り立つため，$ERS_1 \leq ERS_3$ が成り立つ．したがって定義 9 より，$ERG_1 \leq ERG_3$ が成り立つ．

ExPD： ロール $r_2 \in ERS_2$ からダイレクト特権 p を削除してロール $r_3 \in ERS_3$ になったと仮定する．ExPD の適用条件より，r_2 の任意の直シニアロール $sr \in ERS_2$ に対して $p \in Direct(sr)$ が成り立つ．よって，p を r_2 から削除することで ERS_2 と ERS_3 で変化のあるロールは r_2 と r_3 のみであり，$r_3.rpset = r_2.rpset - \{p\}$ が成り立つ．ここで，

- $\Gamma^{-1}_{ERS_1 \to ERS_2}(r_2) \neq \emptyset$ の場合
  任意のロール $r_1 \in \Gamma^{-1}_{ERS_1 \to ERS_2}(r_2)$ に対して，ExPD の適用条件より $p \notin r_1.rpset$ が成り立つ．今，$r_1 < r_2 \wedge p \notin r_1.rpset \wedge p \in r_2.rpset$ であるため，補題 3 より $r_1 \leq r_3$ が成り立つ．すなわち，任意の r_1 に対して $\Gamma_{ERS_1 \to ERS_3}(r_1) \neq \emptyset$ が成り

⋆1 r_3，各 sr_3 と等価なロールが ERS_2 に属する場合，$ERS_2 = ERS_3$ が成り立つ．

立つ．ERS_2 と ERS_3 の違いは r_2 と r_3 のみであり，仮定より任意の $r \in ERS_1$ に対して $\Gamma_{ERS_1 \to ERS_2}(r) \neq \emptyset$ が成り立つため，$\Gamma_{ERS_1 \to ERS_3}(r) \neq \emptyset$ が成り立つ．ここで $ERS_1 \neq ERS_3$ のとき，定義 8 より $ERS_1 < ERS_3$ が成り立つ．$ERS_2 = ERS_3$ の場合も考慮すると，$ERS_1 \leq ERS_3$ が成り立つ．

- $\Gamma^{-1}_{ERS_1 \to ERS_2}(r_2) = \emptyset$ の場合
  仮定より $ERS_1 \leq ERS_2$ が成り立つため，補題 1 より $ERS_1 \leq ERS_2 - \{r_2\}$ が成り立つ．よって，補題 2 より $ERS_1 \leq (ERS_2 - \{r_2\} \cup \{r_3\})$ が成り立つため，$ERS_1 \leq ERS_3$ が成り立つ．

したがって定義 9 より，$ERG_1 \leq ERG_3$ が成り立つ．

ExRA：ERS_2 にロール r を追加して ERS_3 になったと仮定する．仮定より $ERS_1 \leq ERS_2$ が成り立つため，補題 2 より $ERS_1 \leq ERS_3$ が成り立つ．したがって定義 9 より，$ERG_1 \leq ERG_3$ が成り立つ．

ExRD：ERS_2 からロール r_1 を削除して ERS_3 になったと仮定する．ExRD の適用条件より $r_1.rpset = r_2.rpset$ を満たすロール $r_2 \in ERS_2$ が存在する．よって補題 4 より，$ERS_2 = ERS_2 - \{r_1\} = ERS_3$ が成り立つ．仮定より $ERS_1 \leq ERS_2$ が成り立つため，$ERS_1 \leq ERS_3$ が成り立つ．したがって定義 9 より，$ERG_1 \leq ERG_3$ が成り立つ．　□

5. 等価変換操作と拡張変換操作によるロール定義の更新

3.2 節で述べたように，拡張ロールグラフを更新する変換操作は，拡張ロールグラフのアクセス権限を変更しない変換操作（等価変換操作），アクセス権限を拡大する変換操作（拡張変換操作），アクセス権限を縮小する変換操作に分類できる．

本章では，2.2 節で述べた **P1** と **P2** を解決するために，拡張ロールグラフ間の等価関係，拡張関係，および，等価変換操作，拡張変換操作を用いる．まず **P1** の解として，ロール r_1 と r_2 が適切にマッピングされている場合において，r_2 が等価関係，もしくは，拡張関係にあるロール r_3 に更新された場合に，ロールの適切なマッピングが維持されているとする．この定義のもとでは，$r_2.rpset \subseteq r_3.rpset$ が成り立つ．すなわち，更新後のロールのアクセス権限は，更新前のロールのアクセス権限より縮小しないことが保証される．このとき **P2** を解決する更新操作は，3.2 節で述べた 5 つの等価変換操作と 4.2 節で述べた 4 つの拡張変換操作となる．アクセス権限を縮小する変換操作は，上記で定義した **P1** の解を満たさないため，**P2** を解決する更新操作として不適である．

このような定義のもとで，コンピュータ間のロールのマッピング，および，ロール定義の更新は以下の手順で実現される．

ステップ 1：アクセス先コンピュータの管理者が，アクセス先コンピュータ上のロールとアクセス元コンピュータ上のロールをマッピングする．これはロールのマッピングの初期状態に相当する．

ステップ 2：ロール定義の更新には，等価変換操作，もしくは，等価拡張変換操作を用いる．これらの変換操作のみを用いることで，初期状態でマッピングされるロールと比較して等価，もしくは，拡張関係にあるロールへのマッピングが保証される．

最初にコンピュータ間のロールをマッピングするとき，アクセス先コンピュータの管理者がマッピングすることが望ましい．その理由として，(1) オブジェクトはアクセス先コンピュータに存在し，オブジェクトへのアクセス権限を認可するのはアクセス先コンピュータの管理者であること，(2) アクセス元コンピュータと等価なロールがアクセス先コンピュータに存在しないとき，アクセス先コンピュータにおいてサブジェクトに認可するアクセス権限をアクセス元コンピュータより縮小する（特権の少ないロールを割り当てる）か拡大する（特権の多いロールを割り当てる）かを決める必要があること，があげられる．

アクセス先コンピュータの管理者が初期状態で適切にマッピングしたロールに対して，ロール定義の更新後のロールが等価関係，拡張関係を維持することで，ロール定義の更新にともないアクセス権限が縮小しないことが保証される．すなわち，初期状態でアクセス先コンピュータにおいてサブジェクトに認可されたアクセス権限は，ロール定義の更新により取り消されることはない．等価変換操作，拡張変換操作を用いてロール定義を更新することで，ロール定義の更新後のロールがロール定義の更新前のロールに対して等価関係，拡張関係を維持することを保証できる．よって，ロールの適切なマッピングが維持されるため，ロール定義の更新においてロールのマッピングの確認作業が不要となり，アクセス先コンピュータの管理者の負荷を軽減できる．

定理 1 で示した ExPD と ExRD の 2 つの拡張変換操作は，拡張変換を実現するのに必須の変換操作ではない．なぜなら，これらは新たに追加したロールや特権を削除するための変換操作であり，管理者によるロール定義の更新において，不要なロールや不要な特権を追加した場合にのみ用いられる変換操作だからである．しかし管理者は，無駄な更新をすることなく自身の望むロール定義に拡張ロールグラフを更新できるとは限らない．更新の過程において，管理者は更新完了後のロール定義では不要となるロールや特権を追加する場合もあると考えられる．ExPD や ExRD の 2 つの拡張変換操作がない場合，更新の過程で追加し

た不要なロールや不要な特権を削除することができず，ロールや特権の追加はやり直しのできない変換操作となってしまう．このことはロール定義の更新において，管理者に対する大きな制約となる．ExPD と ExRD の 2 つの拡張変換操作の存在意義は，この制約を取り除くことにある．

6. システム適用事例

5 章では **P1** の解と **P2** の解を述べ，ロール定義の更新手順を述べた．本章では，このロール定義の更新手順に基づき，2.2 節で述べたプロジェクト A のファイルサーバを対象とするロール定義の更新の事例を示す．

6.1 ファイルサーバ上のロール定義

ファイルサーバ上に定義されるロールは，ファイルサーバ管理者によりロールグラフとして管理されている．ファイルサーバ上に定義されるロールグラフ ERG を**図 4** の (a) に示す．なお図を簡略化するために，各ロールのダイレクト特権のみを記述し，$MinRole$ と $MaxRole$ は省略している．ファイルサーバでは $MinRole$ と $MaxRole$ 以外に，週報を作成する特権を関連付けられた $ProjMember$，ソースファイルの読み書きと検査ツールおよびコンパイラの使用権限を関連付けられた $SProgrammer$，営業報告書を作成する特権を関連付けられた $SalesStaff$，プロジェクト報告書を作成する特権を関連付けられた $ProjManager$ が定義されている．各ロールのイフェクティブ特権集合を**表 1** の ERG の列に示す．また 5 章で述べたステップ 1 として，ファイルサーバ管理者が，クライアント上のロールをファイルサーバ上のロールに初期状態としてマッピングしている．

6.2 ロール定義の更新

5 章で述べたステップ 2 として，検査要員向けのロール $Tester$ とプロジェクト A の子プロジェクトであるプロジェクト B のプログラマ向けのロール $SProgrammer\_B$ を追加するために，ロールグラフ ERG をロールグラフ ERG' に拡張することを考える．検査要員はソースファイルの読み込みは可能だが書き込みはできず，プロジェクト B のプログラマはプロジェクト B 向けのソースファイルのみ読み書き可能とする．また，プロジェクト A のプログラマはプロジェクト B 向けのソースファイルの読み書きも可能とする．さらに，プロジェクト B はプロジェクト A を担当する会社の子会社が担当し，子会社は検査ツールの使用ライセンスを所有していないとする．

ファイルサーバ管理者によるロールグラフの変換過程を図 4 に示す．図 4 の (a) はロールグラフ ERG であり，(f) は変換後のロールグラフ ERG' である．(a) から (f) に至る各変換過程では，等価変換操作，拡張変換操作を適用している．(a) に ExRA を適用することでロール $Tester$，$SProgrammer\_B$ を追加し，(b) を得る．(b) に ExPA を適用することで $Tester$ に特権 r_src，r_src_B，$use\_profiler$，$use\_compiler$ を，$SProgrammer\_B$ に特権 r_src_B，w_src_B，$use\_profiler$，$use\_compiler$ を関連付け，(c) を得る．(c) に RPD を適用することで $SProgrammer$ から特権 $use\_profiler$，$use\_compiler$ の関連付けを削除し，(d) を得る．

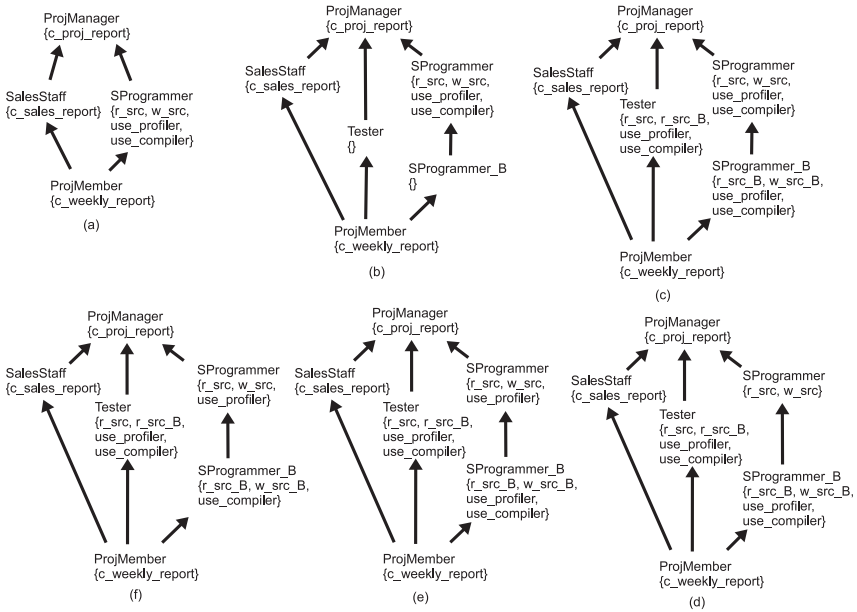

図 4　ロールグラフの変換
Fig. 4　Transformation of a role graph.

管理者がここまでロールグラフを変換したところで，子会社が検査ツールのライセンスを所有していないことを指摘されたとする．そのため，$SProgrammer\_B$ から 1 度関連付けた特権 $use\_profiler$ の関連付けを削除することにした．(d) に ExPA を適用することで $SProgrammer$ に特権 $use\_profiler$ を関連付け，(e) を得る．(e) に ExPD を適用する[⋆1]こ

⋆1 $\Gamma^{-1}_{(a)\to(e)}(SProgrammer\_B) = \emptyset$ であるため，ExPD を適用可能である．

表 1　ロールに割り当てられているイフェクティブ特権集合
Table 1　Sets of effective privilege associated with each role.

Role defined in file server	ERG	ERG′
ProjMember	c_weekly_report	c_weekly_report
SProgrammer_B		c_weekly_report, r_src_B, w_src_B, use_compiler
SProgrammer	c_weekly_report, r_src, w_src, use_profiler, use_compiler	c_weekly_report, r_src, w_src, use_profiler, use_compiler r_src_B, w_src_B
Tester		c_weekly_report, r_src, r_src_B, use_profiler, use_compiler
SalesStaff	c_weekly_report, c_sales_report	c_weekly_report, c_sales_report
ProjManager	c_weekly_report, r_src, w_src, use_profiler, use_compiler, c_sales_report, c_proj_report	c_weekly_report, r_src, w_src, use_profiler, use_compiler, c_sales_report, c_proj_report, r_src_B, w_src_B

とで $SProgrammer\_B$ から特権 $use\_profiler$ の関連付けを削除し，(f) を得る．最終的に得たロールグラフ (f) は，非正規拡張ロールグラフではないため有効である．

ロール定義更新後の ERG' の各ロールのイフェクティブ特権集合を表 1 の ERG' の列に示す．ERG から ERG' への変換において $ProjMember$，$SProgrammer$，$SalesStaff$，$ProjManager$ のイフェクティブ特権集合は縮小していないため，ロールの適切なマッピングは維持される．この事例のように，5 章で述べたロール定義の更新手順では，ロールの適切なマッピングが維持される．

6.3　マッピングの適切さを個々に確認する手法と提案手法の工数の差異

この節では，6.2 節で述べたプロジェクト A を例として，マッピングの適切さを個々に確認する手法（個別手法）と提案手法でのファイルサーバ上のロールグラフ更新（すなわち，ロール定義の更新）時の工数の差異を示す．

プロジェクト A のクライアント上には**表 2** の Role defined in client 列に示すロールが定義されており，各ロールは表 2 の Role set mapped in file server 列に示すファイルサーバ上のロールの集合（このロールの集合をマッピング先ロール集合と呼ぶ）に属する各ロールにマッピングされているものとする．

最初に，個別手法によるマッピング先のロールグラフ更新手順を以下に示す．

個別手順 1：ロールグラフを更新する．

個別手順 2：ロール間のマッピングがロールグラフ更新後も適切であるか確認する．個別手順 2 は以下の 3 ステップに分割できる．

表 2　ロールとマッピング先ロール集合
Table 2　Roles and mapped role sets.

Role defined in client	Role set mapped in file server
LProgrammer	{SProgrammer}
LSalesStaff	{SalesStaff}
LProjManager	{ProjManager}

ステップ 2-1：クライアント上に定義されるすべてのマッピング元のロールに対し，サーバ上に定義されるマッピング先ロール集合を求める．クライアントが複数存在する場合，すべてのクライアントを対象にマッピング先ロール集合を求める．

ステップ 2-2：ステップ 2-1 で求めたマッピング先ロール集合ごとに，ロールグラフ更新前後での，マッピング先ロール集合に属するすべてのロールに関連付けられているイフェクティブ特権集合の和を求める．

ステップ 2-3：ステップ 2-2 で求めたすべてのイフェクティブ特権集合が，ロールグラフ更新後に縮小していないことを確認する．縮小していた場合，個別手順 1 に戻ってロールグラフを再更新する．

本例の個別手法によるファイルサーバ上のロールグラフの更新手順は，上記個別手順に従い以下のようになる．

個別手順 1：図 4 の (a) のロールグラフを以下に示す順序で (f) に更新する．

(1)　$Tester$，$SProgrammer\_B$ を追加する．
(2)　$SProgrammer$ から $use\_compiler$ の関連付けを削除する．
(3)　$Tester$ に r_src，r_src_B，$use\_profiler$，$use\_compiler$ を，$SProgrammer\_B$ に r_src_B，w_src_B，$use\_compiler$ を関連付ける．

個別手順 2：クライアント上のロールとファイルサーバ上のロール間のマッピングが適切であるかを以下に示す 3 ステップで確認する．

ステップ 2-1：すべてのマッピング先ロール集合を求める．その結果を表 2 の Role set mapped in file server 列に示す．

ステップ 2-2：ロールグラフ更新前後での，マッピング先ロール集合に属するすべてのロールに関連付けられているイフェクティブ特権集合の和を求める．その結果を**表 3** の Role graph (a)，(f) 列に示す．

ステップ 2-3：表 3 に示すように，ステップ 2-2 で求めたイフェクティブ特権集合は，

表 3　マッピング先ロール集合ごとのイフェクティブ特権集合
Table 3　Sets of effective privilege of each mapped role set.

Role set mapped in file server	Role graph (a)	Role graph (f)
{SProgrammer}	c_weekly_report, r_src, w_src, use_profiler, use_compiler	c_weekly_report, r_src, w_src, use_profiler, use_compiler r_src_B, w_src_B
{SalesStaff}	c_weekly_report, c_sales_report	c_weekly_report, c_sales_report
{ProjManager}	c_weekly_report, r_src, w_src, use_profiler, use_compiler, c_sales_report, c_proj_report	c_weekly_report, r_src, w_src, use_profiler, use_compiler, c_sales_report, c_proj_report, r_src_B, w_src_B

ロールグラフ更新後に縮小していない．よって，クライアントとファイルサーバ間のロールの適切なマッピングは維持されている．

次に，提案手法によるマッピング先のロールグラフ更新手順を以下に示す．

提案手順 1：ロールグラフを更新する．この際，等価変換操作，拡張変換操作のみを用いて更新する．

等価変換操作，拡張変換操作のみを用いて更新するため，更新前に存在するすべてのロールに関連付けられているイフェクティブ特権集合は更新後において縮小しない．よって，ロールグラフ更新前後においてロールの適切なマッピングは維持されている．

本例における提案手法によるファイルサーバ上のロールグラフの更新手順は，上記提案手順に従い以下のようになる．

提案手順 1：図 4 の (a) のロールグラフを以下に示す順序で (f) に更新する．

（1） ExRA を適用し，*Tester*，*SProgrammer_B* を追加する．

（2） ExPA を適用し，*Tester* に *r_src*，*r_src_B*，*use_profiler*，*use_compiler* を，*SProgrammer_B* に *r_src_B*，*w_src_B*，*use_compiler* を関連付ける．

（3） RPD を適用し，*SProgrammer* から *use_compiler* の関連付けを削除する．

個別手法では，設定の異なるクライアントの数の増加にともないマッピング元のロールの種類が増加し，マッピング先ロール集合の数（マッピング先のロールグラフに含まれるロール数を n とすると，とりうる最大値はロール集合の部分集合の数から空集合を除いた $2^n - 1$）も増加する．その結果，個別手順 2 のステップ 2-2 で求めなければならないイフェクティブ特権集合の和の数も増加し，工数が増大する．また，ロールの適切なマッピングが維持されない場合，個別手順 1 からやり直す必要があり，更新全体の工数は管理者の経験や技量に依存してしまう．それに対し，提案手法は個別手順 2 の工数をそもそも必要とせず，提案手順 1 を 1 度実施するだけで良い．以上より，提案手法はロールグラフ更新の工数を削減できる．

7. おわりに

本論文では，アクセス制御として RBAC を適用する分散システムにおいて，ロールの適切なマッピングの維持を定義し，ロールの適切なマッピングを維持するロール定義の更新操作を定義した．これらを定義するために，拡張ロールグラフ間の拡張関係と拡張関係を維持しながら拡張ロールグラフの変換を可能にする拡張変換操作を提案した．等価変換操作と拡張変換操作を用いてロール定義を更新することで，ロールのマッピングの確認作業が不要となり，システム管理者の負荷を軽減できる．

今後の課題として，ロール定義の更新時に，ロールの適切なマッピングを維持する条件に多様性を持たせることがあげられる．本論文で提案した拡張関係では，ロール定義の更新により，ロールのアクセス権限は縮小しないことが保証される．しかし分散システムによっては，ロール定義の更新により，ロールのアクセス権限が拡大しないことを保証したい場合もあると考えられる．この場合，ロールの適切なマッピングを維持する条件を変更する必要がある．さらに，システム管理者に有用でより複雑な変換操作を提供することがあげられる．本論文で提案した拡張変換操作は，ロールや特権の追加，削除のみを行う単純な変換操作である．しかし単純な変換操作を組み合わせて，より複雑な変換操作を定義することも可能である．このような変換操作を用いることで，ロール定義の更新作業の効率を上げることができる．

参考文献

1) Griffiths, P.P. and Wade, B.W.: An authorization mechanism for a relational database system, *ACM Trans. Database Syst.*, Vol.1, No.3, pp.242–255 (1976).
2) Gustafsson, M., Deligny, B. and Shahmehri, N.: Using NFS to Implement Role-Based Access Control, *Proc. 6th Workshop on Enabling Technologies on Infrastructure for Collaborative Enterprises*, IEEE Computer Society, pp.299–304 (1997).
3) Joshi, J.B.D., Aref, W.G., Ghafoor, A. and Spafford, E.H.: Security models for web-based applications, *Commun. ACM*, Vol.44, No.2, pp.38–44 (2001).
4) Sandhu, R.S., Coyne, E.J., Feinstein, H.L. and Youman, C.E.: Role-Based Access Control Models, *IEEE Computer*, Vol.29, No.2, pp.38–47 (1996).
5) Ferraiolo, D.F., Sandhu, R.S., Gavrila, S.I., Kuhn, D.R. and Chandramouli, R.:

Proposed NIST standard for role-based access control, *ACM Trans. Information and System Security*, Vol.4, No.3, pp.224–274 (2001).
6) Ma, M. and Woodhead, S.: Constraint-Enabled Distributed RBAC for Subscription-Based Remote Network Services, *Proc. 6th IEEE International Conference on Computer and Information Technology*, IEEE Computer Society, p.160 (2006).
7) Tari, Z. and Chan, S.-W.: A Role-based Access Control for Intranet Security, *IEEE Internet Computing*, Vol.1, No.5, pp.24–34 (1997).
8) Freudenthal, E., Pesin, T., Port, L., Keenan, E. and Karamcheti, V.: dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments, *Proc. 22nd International Conference on Distributed Computing Systems*, IEEE Computer Society, p.411 (2002).
9) Nyanchama, M. and Osborn, S.L.: Access Rights Administration in Role-Based Security Systems, *Proc. IFIP WG11.3 Working Conference on Database Security VII*, pp.37–56, North-Holland Publishing Co. (1994).
10) Nyanchama, M. and Osborn, S.L.: The Role Graph Model and Conflict of Interest, *ACM Trans. Information and System Security*, Vol.2, No.1, pp.3–33 (1999).
11) Asakura, Y. and Nakamoto, Y.: Extending a Role Graph for Role-Based Access Control, *IEICE Trans. Inf. & Syst.*, Vol.E92.D, No.2, pp.211–219 (2009).
12) Aho, A.V., Garey, M.R. and Ullman, J.D.: THE TRANSITIVE REDUCTION OF A DIRECTED GRAPH, *SIAM Journal of Computing*, Vol.1, No.2, pp.131–137 (1972).

(平成 21 年 5 月 25 日受付)
(平成 21 年 12 月 17 日採録)

朝倉　義晴（正会員）

1973 年生．1998 年大阪大学大学院基礎工学研究科修士課程修了．同年 NEC 入社．2007 年兵庫県立大学大学院応用情報科学研究科博士課程入学．組み込みシステム，分散システム，情報セキュリティ技術に興味を持つ．電子情報通信学会会員．

中本　幸一（正会員）

1958 年生．1982 年大阪大学大学院基礎工学研究科前期課程修了．同年 NEC 入社．1990〜1991 年 Cornell 大学計算機科学科客員研究員．1997 年大阪大学大学院情報数理系専攻博士課程入学．2000 年単位取得退学．博士（工学）．2003〜2004 年電気通信大学客員教授．2004 年より現職．2006 年より名古屋大学大学院情報科学研究科附属組込みシステム研究センター特任教授，組み込みソフトウェア，モバイルシステムソフトウェア，ソフトウェア工学に興味を持つ．電子情報通信学会，日本ソフトウェア科学会，IEEE Computer Society，ACM，USENIX 各会員．