情報処理学会論文誌 Vol. 49 No. 9 3062–3073 (Sep. 2008)

来歴に基づく マルチレベルセキュリティ文書管理システム

三品拓也[†1] 勝野恭治[†1,†2]
吉濱佐知子[†1] 工藤道治[†1]

提携・合併・買収・アウトソーシングといったビジネス環境の変化と，インターネットのような組織間通信経路の発達により，オフィス文書が組織や会社をまたいでやりとりされる機会が増えている．これにともない悪意のないユーザの誤操作による情報漏洩の可能性が増しており，実際に情報漏洩事故の報告が後を絶たない．このような事故を防ぐために，ユーザの注意力に頼ることなく機密性を確保することが求められている．既存技術であるマルチレベルセキュリティは厳密な情報フロー制御を実現可能であるが，メタ情報欠落問題・機密解除問題という 2 つの実用上の問題があり，オフィス文書管理システムに適用することは困難であった．そこで本論文ではメタ情報欠落問題を解決するため，オフィス文書の来歴を記録して文書に安全な形で添付する来歴封入と，そのデータ構造を提案する．またオフィス文書の機密解除問題を解決するため，文書よりも細かい文書要素の粒度でセキュリティラベルを付与し，セキュリティラベルに基づいて情報フロー制御を行う細粒度情報フロー制御機構を提案する．その際，ラベル付与は来歴に基づいて可能な限り自動化し，ラベル付与のコストを削減する．さらに，来歴封入と細粒度情報フロー制御機構のプロトタイプをそれぞれ ODF（Open Document Format）と OpenOffice.org に実装してその実現可能性を示す．

A Multi-level Security-based Document Management System using Provenance

Takuya Mishina,[†1] Yasuharu Katsuno,[†1,†2]
Sachiko Yoshihama[†1] and Michiharu Kudo[†1]

Current business situations require improved confidentiality and integrity for office documents. The Multi-level Security (MLS) model can provide an information flow control feature to content management systems, however, the meta-information lost problem and the declassification problem prohibit the use of the MLS. In this paper we propose a meta-data format called *sticky provenance* and a fine-grained information flow control system using the sticky provenance. The sticky provenance contains the change history and the labels of an office document in a secure form, and it ensures the confidentiality of the change history of the documents in distributed environments. The fine-grained information flow control system reduces the label creep problem of the information flow control models with the sticky provenance. In other words, the sticky provenance and the fine-grained information flow control system can introduce a practical fine-grained information flow control capability to office applications so that we can ensure the confidentiality of office documents.

1. はじめに

1.1 背景

オフィス文書はビジネスに関する重要な情報を保管・伝達するための主要な手段である．オフィス文書には，組織をまたいでやりとりされることや，文書をまたいで情報が再利用されることが多い，といった特徴がある．提携・合併・買収・アウトソーシングといったビジネス環境の変化とインターネットのような組織間通信経路の発達によって前述の特徴はより際立ってきており，オフィス文書の情報漏洩リスクはますます高まっている．たとえばオフィス文書を添付した電子メールを誤った宛先に送ってしまったり，オフィス文書を意図せず公衆ピア・ツー・ピア（P2P）ネットワークに公開してしまったりした場合，ユーザに悪意がなくても情報漏洩が発生する可能性がある．実際にそのような事故例の報告が近年後を絶たず，ゆえになるべくユーザの注意力に頼らずオフィス文書の機密性を確保することが求められている．

1.2 マルチレベルセキュリティモデル

オフィス文書の漏洩が発生する大きな要因として，オフィス文書管理システムの内部で発生する不適切な情報フローがあげられる．たとえば，アクセス制御機能を持った文書管理システムに機密文書 d_1 と公開文書 d_2 が保管されているものとする．ここで d_1 へのアクセス権を持つユーザが d_1 に含まれる機密情報を公開文書 d_2 にコピーすると，公開文書を通じて機密情報が漏洩してしまう．

このような情報漏洩が発生することを防ぐためのモデルが Bell-LaPadula モデル[1] であ

†1 日本アイ・ビー・エム株式会社東京基礎研究所
IBM Research, Tokyo Research Laboratory
†2 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba

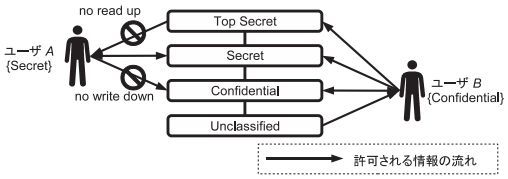


図 1　MLS システムにおいて許可される情報の流れ
Fig. 1　Information flow allowed by MLS.

る．Bell-LaPadula モデルは機密性確保のためのマルチレベルセキュリティ（Multi-level Security; MLS）モデルであり，あらかじめ順序関係を持つセキュリティクラスを定義して，主体（subject）にセキュリティクリアランス，対象（object）にセキュリティラベルを付与し，セキュリティラベルごとに定義されたポリシに基づいて情報フロー制御を行い，機密性を確保するモデルである．Bell-LaPadula モデルが持つ重要な特性の中に，$*$-特性（$*$-property）と呼ばれる特性がある．$*$-特性を満たすシステムでは，あるセキュリティクリアランスを付与された主体は，より低いセキュリティラベルが付与された文書に対する書き込み権限を持たない（no write down）．この特性と，付与されたセキュリティクリアランスよりも高いセキュリティラベルが付与された文書に対する読み込み権限を持たないこと（no read up）を表す ss（simple security）-特性が MLS の特徴である．$*$-特性および ss-特性を備えた MLS システムにおいて許可される情報の流れは**図 1** の矢印で示された経路のみであり，先にあげた例の「アクセス制御機能を持った文書管理システム」が MLS システムである場合，d_2 の機密レベルが d_1 よりも低いため，文書 d_1 にアクセスできる権限を持つユーザは文書 d_2 に書き込みを行うことができない．このような特性によって，Secret 文書の内容がユーザ A と Confidential 文書を経てユーザ B に漏洩する事故を防ぐことができる．

MLS システムは機密性を確保するために非常に有効なシステムである．しかし実ビジネスに利用できるような MLS 機能付き文書管理システムは，メタ情報欠落問題と機密解除問題という 2 つの大きな問題により実現が困難であると考えられている．

メタ情報欠落問題とは，MLS が付与するセキュリティラベルなどのメタ情報がシステム間を伝播する途中で欠落してしまう問題である．メタ情報が欠落すると，情報がシステムの範囲外へ流出した時点で，メタ情報に基づいて行われる情報フロー制御（例：ポリシに従って非正規ユーザのアクセスを拒否する）が行われなくなり，情報漏洩のリスクが増大する．機密性を確保するためには，ユーザは新しいファイルシステム上でも適切にラベルを再付与しなくてはならないが，伝播のたびにラベル再付与を行うことはユーザにとって大きな負担である．メタ情報欠落問題に起因する情報漏洩の例としては，以下のようなシナリオが考えられる．

- 文書共有サーバに「極秘」扱いで登録されている資料をクライアント PC にダウンロードして編集し，部門データベースにコピーした．このとき部門データベースでのアクセス制御を設定し忘れたため，部門に所属する全ユーザが極秘文書にアクセス可能になってしまった．

メタ情報欠落問題は軍事情報システムのように閉じたシステムでは発生しないが，オフィス文書はネットワークを通じて様々なシステムをわたり歩くため，つねに意図どおりのアクセス制御を実施するためにはメタ情報欠落問題を解決する必要がある．

機密解除問題とは，ラベルクリープ[3) [p.280]] を解消するための操作である機密解除（declassification，downgrading）[3) [p.321]] が，ユーザに大きな負荷を与える問題である．ラベルクリープとは，ユーザがある文書に対して情報フローを繰り返すことで，当該文書に付与されているセキュリティラベルが文書の内容から見て適切と考えられるセキュリティラベルよりも高い機密度になってしまう状態をいう．ラベルクリープが発生すると，本来許可されるべき情報フローまで遮断されて業務遂行に支障が生じるため，ラベルクリープを解消するためにセキュリティラベルを内容に即して付与し直す例外的操作，すなわち機密解除操作を適切に実施する必要がある．機密解除を実行してよいのかどうか判断するのは，機密解除を実行する者（trusted subject $^{\star 1}$）にとって大きな負担である．実際の場面で trusted subject が機密解除を実施しようとした場合，対象文書の内容を隅々まで確認しなければならないからである．オフィス文書でいえば，それに含まれる大量のパラグラフ・図表・画像を洩れなくチェックする必要があるため，ユーザの不注意によって情報漏洩が発生する可能性がある．例として以下のようなシナリオが想定される．

- コンサルテーション資料を再利用して営業用資料を作成しようと考えた．その際，重要プロジェクト資料からお客様の名前などの機密保持すべき情報を削除する必要があったにもかかわらず，一部の記述がチェックをすり抜けてしまい，営業用資料に顧客情報が紛れ込んでしまった．

実ビジネスにおいて MLS を利用する場合は，情報漏洩の防止が機密解除実行コストより

$\star$1 MLS システムでは，MLS に拘束される一般ユーザとは異なり，機密解除操作を実行できるユーザを trusted subject として特別に定義している．

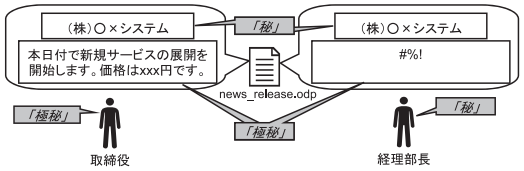


図 2　細粒度ラベリングを行い，ユーザのセキュリティクリアランスに従って情報フロー制御を行った場合のオフィス文書の外観例

Fig. 2　An example appearance of an office document under the control of information flow control feature with fine-grained labeling.

も重要なシステム（例：軍事用途）とは異なり，業務効率と利便性を確保するために，ある程度のリスクは許容して機密解除を許可せざるをえない．しかし，ユーザにとって機密解除の判断を下すために必要な労力は大きいので，これを緩和しなければ，業務効率が大幅に低下したり，不注意による誤った機密解除を行ってしまう可能性がある．

1.3　本論文の貢献

本論文では，実用的な文書管理システムに MLS を適用する際に生じるメタ情報欠落問題および機密解除問題を解決するための手法と，オフィス文書処理アプリケーション（以下「オフィスアプリケーション」）において当該手法を実施するためのデータ構造および情報フロー制御機構を提案する[⋆1]．

まずメタ情報欠落問題を解決するための方法として，メタ情報を文書そのものに添付する「来歴封入」を提案する．来歴とは，内容・セキュリティラベル・その他の変更履歴を安全な形で記録したものであり，来歴封入の考え方は，ポリシをデータに添付する sticky policy[6),7)] の考え方を，一般的なメタ情報に拡張したものである．データにメタ情報を添付するためには，メタ情報が削除されたり改ざんされたりしないような仕組みがデータに備わっていなければならないので，来歴封入のために新しいデータ構造を提案する．

次にオフィス文書に対する機密解除問題を解決する手段として，来歴に基づく細粒度情報フロー制御を提案する．細粒度情報フロー制御とは，**図 2** にあるように，セキュリティラベルを文書単位ではなく，もっと細かい粒度，具合的には段落・図形などといった文書要素（コンポーネント）に対して付与し（細粒度ラベリング），それに基づいて情報フロー制御を行う仕組みをいう[⋆2]．細粒度情報フロー制御では機密解除実行時にシステムが自動的に機密度の高い文書要素をユーザに提示することができるので，ユーザは提示された要素だけを検査すればよくなり，結果として機密解除のコストを低く抑えることができる．ただし，細粒度ラベリングにはユーザがラベルを適切に付与するコストが増えるという課題がある．そこで封入した来歴情報を使ってラベルの付与をできるだけ自動的に行うことでこれを解決する．

最後に，提案手法が実際のオフィスアプリケーションおよびオフィス文書に適用可能であり，メタ情報欠落問題・機密解除問題を解決できることを示すため，来歴封入データ構造をオフィス文書の標準形式 Open Document Format（ODF）上で実現し，来歴に基づく細粒度情報フロー制御を実施するプロトタイプをオフィスアプリケーション OpenOffice.org 上に実装し，性能評価を行って，オーバヘッドが実用上問題のない範囲であることを示した．

本論文の構成は以下のとおりである．2 章では，情報フロー制御機構とデータ構造の設計を行う．3 章では 2 章での設計をもとに実際のオフィスアプリケーションおよびオフィス文書に提案手法を実装する．4 章で関連研究について述べ，5 章で結論および今後の課題について述べる．

2.　システム要件

1 章で示した目的を満たすためのシステムとして，オフィスアプリケーションにリファレンスモニタを導入し，機密解除問題を軽減した MLS ベースの文書管理システムを設計する．本章では，設計に先立って前提条件および安全性要件を設定する．

2.1　前 提 条 件

提案するシステムは，主体・対象・操作および計算機環境に関して以下のような前提条件が満たされているものとして設計する．

主体　ユーザは何らかの組織に所属しており，その組織内ではユーザ認証・ロール認可・セキュリティポリシ定義（ロールとセキュリティクリアランス・機密解除権限のマッピング）を行うディレクトリサービスと，すべてのユーザと信頼関係が確立していて，機密度別の鍵を管理・提供する鍵配信機構が整備されている．鍵配信機構はユーザの持つセキュリティクリアランスに応じて，特定の機密度の情報を解読するための対称鍵を安全

⋆1 本論文の概要は The Second International Workshop on Security（IWSEC2007）[5)] において発表している．

⋆2 提案手法におけるセキュリティラベルはコンポーネントに付与されるので，一般的な文書管理システムとは異なり，文書はセキュリティラベルを持たず，単なるコンテナとみなされる．

な形でユーザに配信する．

環境　リファレンスモニタはオフィスアプリケーション内部に導入され，以下で述べる操作・対象を制御する．オフィスアプリケーションおよびその下のソフトウェアスタック（例：共有ライブラリ，オペレーティングシステム，BIOS）は信頼できる基盤ソフトウェア（Trusted Computing Base）である．また，仮想計算機やデバッガを用いてメモリを直接読み書きする攻撃は存在しない．

対象　アクセス制御は文書内のコンポーネントを単位とする．1 つの文書にはラベルの異なる複数のコンポーネントを配置することができる．ただし，コンポーネントとコンポーネントの合併操作（例：複数の段落を 1 つの段落へ集約する操作）は起こらない．

操作　対象（コンポーネント）に対して行われる操作は新規作成・コピー・ペースト・削除・セキュリティラベル変更であり，制御するのはオフィスアプリケーション上で発生する操作のみである．また，文書に対して行われる操作は新規作成・ファイル保存・ファイル読み込みであり，それ以外の手段でオフィスアプリケーションと直接起こる情報フロー（例：画面キャプチャ）は存在しない．

2.2　安全性要件

上記の前提をふまえて，目的を満たすための安全性要件をまとめる．なお，以下では順序関係を持つセキュリティクラス（以下「クラス」）の集合 $\mathcal{C}$ が定義されていて，セキュリティクリアランス C_j を持つユーザを U_j とする（$\leq$，$<$ は右辺が高いクラスであることを表す）．

機密性要件 (1) ラベル管理要件　セキュリティラベル（以下「ラベル」）は文書内の各コンポーネントに付与される．ユーザ U_j が入力するコンポーネントに与えられるクラスの初期値は C_j である．ラベルは $C_k \in \{C_k | C_j \leq C_k\}$ なるラベル C_k に変更することができるが，$C_i \in \{C_i | C_i < C_j\}$ なるラベル C_i に変更することができるかどうかは U_j が持つ機密解除権限に依存する．

機密性要件 (2) アクセス制御要件　U_j が読み込み可能なコンポーネントは $C_i \in \{C_i | C_i \leq C_j\}$ なるラベル C_j を持つコンポーネントのみである（ss-property）．U_j はすべてのコンポーネントに書き込み可能であり，$C_i \in \{C_i | C_i < C_j\}$ なるラベル C_i を持つコンポーネントに書き込みを行った場合，リファレンスモニタ（以下「モニタ」）はそのコンポーネントのラベルを C_j に変更する（*-property）．

分散管理要件　モニタが導入されていない環境では，コンポーネントに対する読み書き操作を実行することができない．モニタが導入されている環境であれば，いったんモニタの防護区域外を経由した情報に対しても，機密性要件に従ってアクセス制御を実現できる．

検証性要件　リファレンスモニタは，あるコンポーネントおよびそのセキュリティクラスの変更履歴をすべて記録し，いつでも検証可能であり，改ざん・削除を検出できる．機密解除を実施した場合は，誰がいつ実行したのかを後日検証可能である．

3. 設　　計

3.1　来　　歴

「来歴（provenance）」という単語の辞書的定義は「あるものの起源，もしくは知りうる限り最も古い履歴」「美術品やアンティークの所有者記録で，権威や品質を見定めるためのガイドとして利用されるもの」とされる[⋆1]．計算機科学，特に e-science と呼ばれる分野では，「データベースに到着したデータ断片の，起源および到着に至った経緯の記述[8]」という定義で第三者がある科学技術計算の結果を検証するために用いられる[9]．

我々は提案手法を実現するにあたり，来歴を「内容，セキュリティラベル，その他の変更履歴を安全な形で記録したもの」と再定義して利用する．本論文で従来用いられているものと異なる定義を採用した理由は，一般的な定義における来歴が主に検証可能性を提供するのに対して，提案手法では機密性を提供することを主眼としているからである．来歴を安全・確実に流通させることで，1.2 節で取り上げたメタ情報欠落問題・機密解除問題を解決することができる．

3.2　来歴封入

メタ情報欠落を起こすことなく来歴を流通させるための手段として，取得した来歴をデータそのものに添付する方法が考えられる．通常の MLS システムでは，メタデータは MLS システムのアクセス制御下におかれて改ざんなどの攻撃から保護されるが，メタ情報をオフィス文書に添付する場合はアクセス制御ではなく暗号化によって保護する必要がある．そこで，オフィス文書に対して来歴情報を暗号学的に紐づける「来歴封入（sticky provenance）」の概念を提案する．封入された来歴はオフィス文書の変更履歴とセキュリティラベルを安全に格納しており，どのシステム上に流通したとしても，オフィス文書に対してセキュリティポリシを強制することができる．

本論文では以下の 2 つの手法を組み合わせて実現される来歴構造を設計する．

- 内容（content）部分に含まれる機密情報をメタ情報部分に移動し，移動した情報と過

⋆1 Oxford Dictionary of English, 2nd Edition, 2003 より．

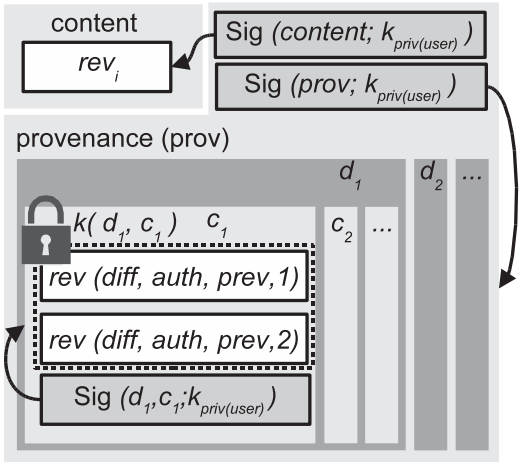


図 3　封入した来歴のデータ構造
Fig. 3　The structure of sticky provenance.

去の編集履歴とあわせて「来歴」とする（移動した情報の代替としてマスク文字列を埋め込む）．

- 来歴をセキュリティラベル単位で分割し，「封印」と署名を行う．

セキュリティラベルは文書が属するカテゴリを表すドメインと文書の機密度の二つ組とする．また「封印」とは，セキュリティラベル固有の暗号鍵によって暗号化を実施することをいう．封印機能により，オフィス文書内の機密情報が認可されていないユーザに漏洩することを防ぐ．また，封印および署名のパフォーマンス・オーバヘッドを最小化するため，暗号化処理は各コンポーネント単位ではなく各セキュリティラベル単位で行う．以上のような，オフィス文書を安全に流通可能な形に変換する処理を「来歴封入処理」と名付け，逆の操作を行って，元の内容部分を復元する処理を「来歴開梱処理」と名付ける．

提案する来歴データの構造を**図 3** に示す．来歴（provenance）の内部はドメインごと（d_i）・機密度ごと（c_i）に分割されたコンテナ群と，来歴部分・内容（content）部分への署名からなり（ただし $k_{priv}(user)$ は最終更新者の秘密鍵），来歴部分・内容部分の改ざんを検出可能とする．1 つのコンテナには複数の変更履歴（rev）が含まれ，1 つの変更履歴は差分（$diff$）・認証情報（$auth$）・1 つ前の変更履歴への参照（$prev$）とリビジョン番号からなる．コンテナはドメイン・機密度に固有の対称鍵（$k(d,c)$）で暗号化され，署名（$\mathrm{Sig}(d,c;k_{priv(user)})$）が付与される．復号時は対称鍵 $k(d,c)$ を後述するドメインサービスから取得して復号を行う．なお，署名は暗号化後のコンテナに付与されるので，そのコンテナの暗号化を解除できないユーザであってもコンテナが改ざんされていないことは検証できる．

来歴封入を行うことによって，オフィス文書の可搬性を確保したまま，メタ情報を欠落させることなく，安全に流通させることができる．ユーザが来歴封入オフィス文書を別のユーザに渡したいときは，従前どおりオフィス文書だけを送付すればよい．その際誤って本来と異なる受信者にオフィス文書がわたってしまったとしても，情報漏洩は発生しない．来歴管理機能のないオフィスアプリケーションで開いた場合は文書全体の来歴開梱処理を行うことができず，来歴管理機能のあるオフィスアプリケーションで開いた場合でも，受信者が閲覧権限を持たない文書要素は対称鍵を取得できないため開梱不可能だからである．逆にいえば，受信者が閲覧権限を持つ文書要素は開梱することができるので，1 つのオフィス文書を流通させるだけで，機密性を損なうことなく，ユーザごとに適切な情報を開示できる．

3.3　細粒度情報フロー制御機構

提案手法では細粒度情報フロー制御機構をオフィスアプリケーション内に追加する．この機構が行う情報フロー制御の特徴は，情報フロー制御を行うタイミングをオフィスアプリケーションからユーザへの情報フローが試みられたとき（文書閲覧時）のみとし，それ以外の情報フローはつねに許可するという点である．すなわち，文書をファイルとして保存するときや文書の一部をクリップボードにコピーするときには，情報フローの許可・遮断を行うのではなく 3.2 節で述べた来歴封入を行う．

細粒度情報フロー制御機構が行うオフィス文書閲覧・編集時の処理を以下で説明する．まず文書閲覧時には，ユーザの認証・認可を確認し，オフィスアプリケーションが行う通常の処理（ファイルの直列化復元）の前に来歴開梱処理を行う．3.2 節で述べたように，来歴はセキュリティラベルごとに異なる鍵で暗号化されているので，ある要素を正しく開梱できるかどうかは，ユーザがそのセキュリティラベルの情報にアクセスする認可を持っているかどうかに依存する．復号できない場合は内容部分の文字列，すなわちポリシで指定されたマスク文字列がそのまま表示される．

続いて，ユーザからの入力にラベルを付与する．ラベルを付与する対象は，オフィスアプリケーションが認識できる最小単位の要素とする．どのようなラベルを付与するのかは，ユーザの認証・認可の状態から判断する．また，すでに付与されているラベルを変更することもできる．ただし機密度が下がる方向のラベル変更は「機密解除」操作であるので，ポリシで認められたユーザにのみ特別なユーザインタフェースを用意して許可する．

細粒度情報フロー制御機構が利用するセキュリティポリシは，大きく分けてドメイン参加ポリシと情報フローポリシからなる．ドメイン参加ポリシとは，ディレクトリサービスによって認証されたユーザもしくはグループに対してどのような権限セットを認可するのかを決めるポリシであり，たとえば「経理部」というドメインがあるとすると，「経理部員は取扱注意（c_3），経理部長は秘（c_2），取締役は極秘（c_1）」といった値を定義できる．一方情報フローポリシはどの権限セットであればどのような操作（読み込み・書き込み・機密解除）が可能であるのかを決めるポリシであり，それらのポリシからたとえば**表 1** のようなアクセス制御行列を得ることができる．

以上のような制御を行うために必要なユーザ定義・ポリシ定義は，細粒度情報フロー制御機構自ら管理するのではなく，任意のディレクトリサービスから取得する．

最後に，提案手法においてファイルやクリップボードを使って情報が移動する様子を**図 4** と**図 5** に示す．リファレンスモニタが導入されているアプリケーションどうしで情報をやりとりする場合は，付与したセキュリティラベルはデータに乗って運ばれ，移動先で復元することが可能である．

表 1　情報フロー制御ポリシから得られるアクセス制御行列の具体例（r：読み込み，w：書き込み，d：機密解除）
Table 1　An example access control matrix given by information flow policy.

	対象の機密度		
主体の機密度	c_1：極秘	c_2：秘	c_3：取扱注意
c_1：取締役	rwd	rd	rd
c_2：経理部長	w	rwd	rd
c_3：経理部員	w	w	rw
ドメイン外	–	–	w

3.4　ユースケース

設計したシステムが安全性要件を満たすことを示すため，以下にユースケースを記述する．なお，ロール「取締役」は「極秘」クリアランスを，ロール「経理部長」は「秘」クリアランスを持つものと仮定する．

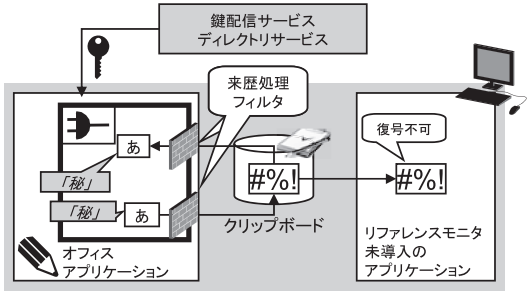


図 5　クリップボードデータに対して来歴封入を行った場合のデータの流れ
Fig. 5　Information flow of clipboard data with sticky provenance.

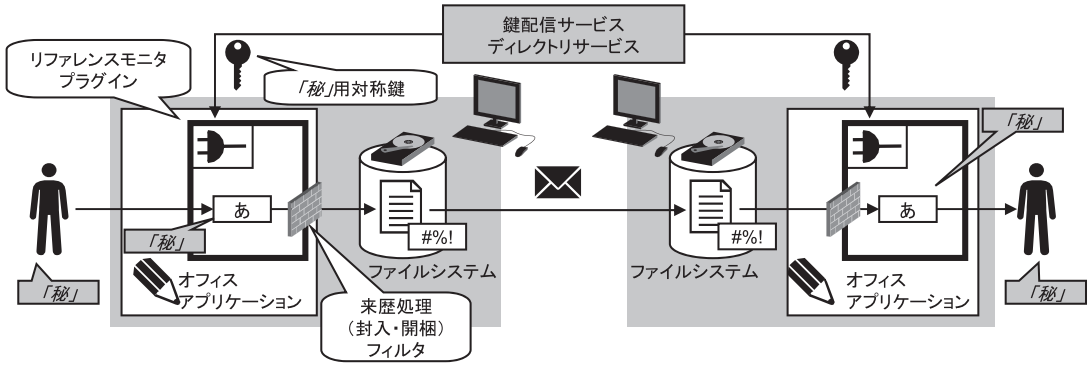


図 4　ファイルに対して来歴封入を行った場合のデータの流れ
Fig. 4　Information flow of office documents with sticky provenance.

(1)　取締役は「極秘」ラベルの付いた社内文書をもとにして営業用資料を作成する．
- 社内文書からのコピー&ペーストを行うとき，各コンポーネントのラベルは，新しい文書に伝えられる（分散管理要件）．
- 誤ってテキストエディタに情報をコピー&ペーストしても，内容が暗号化されているため解読不能である（同上）．
- 情報の中で経理部長に開示可能な情報を選び，当該情報のラベルを機密解除操作によって「極秘」から「秘」に変更する．ラベルがコンポーネントについているので，「極秘」ラベルのコンポーネントだけを確認すればよい（ラベル管理要件）．

(2)　細部の修正を依頼するため，経理部長にメールで転送しようとしたところ，誤って外部の報道機関に送出してしまった．
- 情報は暗号化されているため，報道機関はニュースリリースの中身を解読できない（分散管理要件）．

(3)　経理部長はメールを受信して，編集を開始する．
- 経理部長は出社して，自席の業務用 PC にディレクトリサービスを使ってログインし，メールから添付書類を取り出す．暗号化されたコンポーネントの復号が行われ，編集が可能になる．ただし，経理部長は文書中の「極秘」情報にはアクセスできない（アクセス制御要件）．
- 経理部長が内容を更新することによって，編集したという事実が来歴として記録される（検証性要件）．

(4)　経理部長は編集した文書を再度メールで取締役に送信したが，メールの配送経路上で何者かが文書から経理部長が行った編集内容を一部だけ削除した．
- 取締役は文書の改ざんを検出することができ，経理部長にメールの再送を依頼することで問題なく業務を継続できる（検証性要件）．

4. 実　　装

本論文では，来歴封入を Open Document Format（ODF）[11] に適用し，細粒度情報フロー制御機構を OpenOffice.org [12] 上に実装した．ODF は国際標準化機構で承認された[13] オフィス文書の標準フォーマットであり，公的機関での採用が多く，また多くのベンダがサポートを表明している．OpenOffice.org はオープンソース・マルチプラットフォーム・多言語対応のオフィスアプリケーションであり，ODF ファイルを編集するための実質的なリファレンス実装である．

```
<office:document xmlns:office="http://openoffice.org/2000/office"
 xmlns:prov="http://www.trl.ibm.com/provenance/2007/02">
 <office:meta>
   <prov:Provenance>...</prov:Provenance>
 </office:meta>
 <office:automatic-styles>
   <style:style style:name="gr1">
     <style:properties
      prov:componentId="18b9f214-0467-4542-ac7e-035b016d0934" />
   </style:style>
 </office:automatic-styles>
 <office:body>
   <draw:page>
     <draw:rect draw:style-name="gr1"/>
   </draw:page>
 </office:body>
</office:document>
```

図 6　来歴封入 ODF の構造例

Fig. 6　An example of the ODF file containing the sticky provenance data as a `prov:Provenance` element.

4.1 来 歴 封 入

ODF ファイルは複数の XML ファイル（内容部分である `content.xml`，メタ情報部分である `meta.xml` など）と，画像などのバイナリファイルを zip 形式でひとまとめにしたものである．段落・図形といったコンポーネントは，標準化された XML の要素として表現される．

本論文では 3.2 節で提案した来歴封入データ構造を，ODF に適用するため「来歴封入 ODF（ODFP）」と呼ばれる拡張形式を定義し，図 3 で “provenance” となっている部分を，**図 6** のように ODF の `meta` 要素の子要素 `prov:Provenance` として追加した．各文書要素（図 6 の例では多角形オブジェクト `draw:rect`）にはユーザ定義属性（UserDefinedAttributes）として `prov:ComponentId` 要素を追加し，`prov:Provenance` 要素内の情報と実際の文書要素を紐づけている．`prov:Provenance` 要素は**図 7** のような形をとり，図 6 の各要素が XML 要素として表現されている．なお，図 7 は封印を行う前の構造を示しており，封印実施後は `prov:ClassifiedProvenance` 要素が XML 暗号化[14] で定義されている `EncrytedData` 要素に置き換えられる．

なお，次節で述べるように，来歴封入・来歴開梱処理はファイルだけでなくクリップボードに対しても実施する．その際は来歴封入 ODF の来歴部分のデータ構造をそのまま利用する．

4.2 細粒度情報フロー制御機構

3.3 節で提案した情報フロー制御機構を OpenOffice.org プラグインとして Java™ 言語で実装し，OpenOffice.org 2.3 に導入した．

まず来歴封入・来歴開梱（3.2 節）を行うために，独自の入出力フィルタを実装した．具

```
<prov:Provenance
    xmlns:prov="http://www.trl.ibm.com/provenance/2007/02"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  <prov:DomainProvenance>
    <prov:Domain
     domainId="dom1" uri="http://www.example.com/domain1/">
      <prov:DomainName>domain1</prov:DomainName>
      <prov:DomainDescription>Domain 1</prov:DomainDescription>
      <prov:DomainPolicy>...</prov:DomainPolicy>
    </prov:Domain>
    <prov:ClassifiedProvenance confidentiality="CONFIDENTIAL">
      <prov:ProvenanceEntry
       updateDateTime="2007-03-01T20:46:07.921+09:00">
        <prov:AuthInfo>
          <prov:UserName>tmishina</prov:UserName>
          <prov:RoleName>member</prov:RoleName>
          <prov:Authority type="directory"
           uri="ldap://example.com/" />
        </prov:AuthInfo>
        <prov:Difference>
          <prov:Component
           componentId="18b9f214-0467-4542-ac7e-035b016d0934">
            ...
          </prov:Component>
        </prov:Difference>
      </prov:ProvenanceEntry>
      <ds:Signature>...</ds:Signature>
    </prov:ClassifiedProvenance>
  </prov:DomainProvenance>
  <ds:Signature>...</ds:Signature>
</prov:Provenance>
```

図 7　来歴部分の詳細例

Fig. 7　An example of the sticky provenance in XML.

体的には，通常の ODF 入出力フィルタに SAX（Simple API for XML）フィルタを追加し，このフィルタを使って ODF の XML 処理をフックすることで，ファイル保存時は来歴封入処理を，ファイル読み込み時は来歴開梱処理を実行する．

来歴封入・来歴開梱処理は，ファイル入出力時に加えてクリップボード入出力（コピー・ペースト）時にも行う．たとえばテキストを内包する四角形オブジェクトをコピーする場合は，ユーザからコピー指示を受けた瞬間に，四角形オブジェクトに対して来歴封入を行ってクリップボードにデータを転送する[⋆1]．

5. 実　　験

実装したシステムの実用性を検証するため，実際の環境において文書を編集してパフォーマンスの変化を測定する実験を行った．実験は以下のような条件で行った．

- IBM® ThinkPad T43，Intel® Pentium® M プロセッサ 760（2 GHz），主記憶 1.5 GB
- OpenOffice 2.3（英語版），IBM Java Runtime Environment 1.5.0
- ODF ファイルは zip 圧縮なし・単一 XML 形式
- 各項目 10 回測定し，上限 1 回・下限 1 回を除いた 8 回の平均を結果とする

実験結果を**図 8** に示す．図は縦軸が処理に要する時間であり，横軸には操作の種類および文書内に含まれるコンポーネントの数を示している[⋆2]．まず (A) より，提案手法においてはファイル新規作成時のオーバヘッドがやや大きいことが分かる．これは新規作成時に来歴 DOM 木の新規生成やコンポーネントへのセキュリティラベル付与を行うためである．ただし，内包するコンポーネント数にかかわらず，処理時間はつねにベースライン（来歴処理なし）の約 3 倍となっている．一方 (B)，(C) より，いったん保存したファイルを編集する際のオーバヘッドは比較的小さいことが分かる．また (D) より，クリップボード操作時の来歴処理オーバヘッドはベースラインに比べてやや大きいことが分かるが，絶対的な処理時間としては 1 秒未満であり，ユーザに与える知覚的影響は少ない．

以上より，提案手法は一部の操作でオーバヘッドがかかるものの，頻繁に行われるファイルの編集操作におけるオーバヘッドは少ない．アプリケーションの性質上，ユーザインタラクションに多くの時間を必要とするため，提案手法のオーバヘッドは実用上許容範囲内であると考える．

6. 関 連 研 究

エンタープライズコンテンツ管理システムやデジタル著作権管理システムをはじめとするコンテンツ管理システム（Content Management System; CMS）は，文書もしくはそのポリシを中央集権的サーバで管理し，文書の情報フローを制御するシステムである．CMS は 1 つの組織がつねに所有しているか，もしくはあまり高頻度で更新されない情報を管理するのに適している．一部の製品には，組織外へ流出した文書に対してもセキュリティポリシを強制できるものもある．ただし，MLS の採用例は現在のところ見当らない．

我々がターゲットとしているオープンな環境において，従来のアクセス制御モデルがどのような技術的問題点をかかえているかについては Tolone らの論文[15] に詳しい．また，Bertino ら[16] は，XPath によって複数の組織間で交換される XML 文書に対するセキュリティ手法を提案している．彼女らの提案するインフラストラクチャとデータ構造は，我々の

⋆1 攻撃者に対してどのような文書要素が流れているのかまったく分からない状態にするには，クリップボードに出力するバイト列に対して来歴封入処理を行う必要があるが，今回は OpenOffice.org の実装上の制限から，文書要素に含まれるテキスト要素の内容に対して来歴封入処理を行い，コピー処理を呼び出すことで対応した．前述の例でいえば，文字列を内包する四角形オブジェクトをコピーした場合は内部の文字列のみが暗号化され，四角形そのものの存在は隠蔽されない．

⋆2 来歴封入後のデータサイズはコンポーネント数（1，64，128）によって異なり，順に 2,293 バイト，22,722 バイト，43,459 バイトであった．

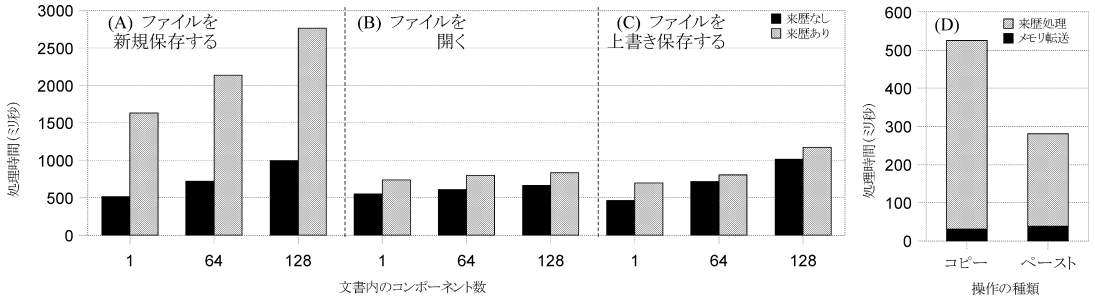


図 8　実験結果
Fig. 8　Experimental results.

提案手法と同様に文書の機密性を確保するものであるが，パスベースのアクセス制御を行っている点が我々の手法と異なる．オフィス文書は頻繁に構造が変わるため，提案手法ではリソースの移動に対応しやすいラベルベースの情報フロー制御を採用した．Pan ら[17] は，中央集権的データベースに保存されているデータを複数組織で共有するためのアクセス制御用ミドルウェアを提案している．また Jin ら[18] も分散環境下で科学技術データを安全に共有するためのフレームワークを提案している．

3.1 節で述べたように，来歴はこれまで主に e-science の分野で用いられてきた概念である．e-science における実験結果の検証にどのようにして来歴を用いるのかという点に関しては，いくつかの議論[9),19),20)] や，具体的な研究が存在する．最近注目されている研究分野として，更新操作[21),22)] やワークフロー統合[23)] といった分野があげられており，オフィス文書のように更新操作が多い対象にこれらの成果を応用することが考えられる．

情報フロー制御のためのフレームワークは近年も活発に研究がなされている分野である．SELinux[2)] はオペレーティング・システム向け情報フロー制御機構の代表例であり，MLS 機能も実装している．また，提案手法よりさらに細粒度なプログラム言語レベルでの情報フロー制御[24)–26)] が近年活発に提案されており，この分野では機密解除問題についても研究が進められているので[27)–30)]，これらの成果を提案手法に取り込むことも考慮する必要がある．

XML アクセス制御もまた活発に研究が行われている分野である[31),32)]．本論文では言及しなかった複数組織間でのポリシ統合について Mazzoleni ら[33)] は，複数の組織が共同で業務を行えるよう，XACML で記述された複数のポリシを統合するアルゴリズムを提案している．

版管理された XML に対してアクセス制御を実施する手法もいくつか提案されており[34),35)]，Iwaihara ら[36)] は XPath を拡張した XVerPath と呼ばれる表現手法を導入して，版の概念を含む XML 要素へのアクセス制御を提案している．また，Damiani ら[37)] は，XML 文書に対する細粒度アクセス制御を提案している．

我々の提案は機密性を主眼に置いているが，実用にあたっては検証可能性の確保も重要である．その際，来歴はセキュリティラベルごとに分割されているため，部分に対する検証可能性が必要となる．このような「部分に対する検証可能性」を実現する手法として墨塗り署名[38)] があげられる．また，本論文では脅威モデルとして「認可されたユーザは悪意を持たない」という仮定を置いているが，認可されたユーザが変更履歴を改ざんする可能性を考慮すべき状況もありうる．その場合はヒステリシス署名[39)] を使うことで変更履歴全体を検証可能となる．

提案手法が実用的なパフォーマンスで動作するのかどうかは，非常に重要な要素である．効率的な差分検出技術[40)] や，アクセス制御のパフォーマンスを改善する技術[41)]，変更検出技術[42)] を活用して，ストレスのない操作感を提供する必要がある．

本論文ではメタ情報欠落問題に対処する方法としてデータにメタ情報を添付する方法をとったが，通信路にメタ情報を添付する方法もある．Jaeger ら[43)] は IPSec の IP パケット（トンネル）にセキュア OS のセキュリティラベルを付与する手法を提案している．

7. ま と め

本論文では既存の MLS システムが抱えるメタ情報欠落問題と機密解除問題を解決するた

め，来歴封入データ構造および来歴に基づく細粒度情報フロー制御機構を設計した．また，それらが実際に利用可能であることを確認するため，ODF および OpenOffice.org に対して実装を行い，性能評価を行って，オーバヘッドが実用上問題のない範囲であることを示した．

本論文では細粒度情報フロー制御機構の設計に際して，ユーザの PC 環境には悪意のあるソフトウェアが存在しないことを仮定しているが，実際にはユーザの PC に悪意のあるソフトウェアが混入する可能性がある．そのため，ユーザの環境が機密情報を操作するのにふさわしいかどうか判断する構成検証技術[44] が重要である．構成検証の結果をポリシの述語として利用することで，たとえばセキュリティパッチを正しく適用している社内の PC からは機密度の高い文書にアクセス可能で，安全性が確認できない自宅 PC からは機密度の低い文書のみアクセス可能にする，といったように，安全性と利便性をより高いレベルで両立させることができる．また，鍵管理に関しては既存の分散環境対応フレームワーク[45),46)] の鍵管理機能を利用することを検討する．

機能拡張としては，オペレーティング・システムレイヤに実装されたリファレンスモニタとの連携があげられる．アプリケーションレイヤのリファレンスモニタは，細粒度で制御を行うことができる反面，システム全体の動作を制御することができない．たとえば「この情報を操作している場合は画面キャプチャと印刷をともに禁止する」といった，オペレーティング・システムの動作を制御するようなポリシを実現するためには，前述の SELinux のようなオペレーティング・システムレイヤのリファレンスモニタと連携できることが望ましい．

謝辞　本研究は，経済産業省，新世代情報セキュリティ研究開発事業の研究として行われたものである．

参 考 文 献

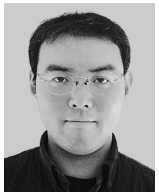
1) Bell, D.E. and LaPadula, L.J.: Secure computer system: Unified exposition and Multics interpretation, Technical Report MTR-2997 Rev.1, MITRE Corporation (1976).
2) Loscocco, P. and Smalley, S.: Integrating Flexible Support for Security Policies into the Linux Operating System, *Proc. FREENIX Track: 2001 USENIX Annual Technical Conference* (2001).
3) Denning, D.E.R.: *Cryptography and Data Security*, Addison-Wesley (1982).
4) Sabelfeld, A. and Sands, D.: Declassification: Dimensions and Principles, *Journal of Computer Security* (2007).
5) Mishina, T., Yoshihama, S. and Kudo, M.: Fine-grained Sticky Provenance Architecture for Office Documents, *the International Workshop on Security 2007* (2007).
6) Karjoth, G., Schunter, M. and Waidner, M.: The Platform for Enterprise Privacy Practices – Privacy Enabled Management of Customer Data, *2nd Workshop on Privacy Enhancing Technologies* (2002).
7) Mont, M., Pearson, S. and Bramhall, P.: Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services, *Proc. 14th International Workshop on Database and Expert Systems Applications* (2003).
8) Buneman, P., Khanna, S. and Tan, W.-C.: Data Provenance: Some Basic Issues, *the 20th Conference on Foundations of Software Technology and Theoretical Computer Science* (2000).
9) Simmhan, Y.L., Plale, B. and Gannon, D.: A survey of data provenance in e-science, *SIGMOD Record*, Vol.34 (2005).
10) OASIS eXtensible Access Control Markup Language (XACML) TC. http://www.oasis-open.org/committees/xacml/
11) OASIS Open Document Format for Office Applications (OpenDocument) TC. http://www.oasis-open.org/committees/office/
12) OpenOffice.org. http://www.openoffice.org/
13) ISO/IEC 26300:2006: Open Document Format for Office Applications (OpenDocument) v1.0 (2006).
14) XML Encryption Syntax and Processing. http://www.w3.org/TR/xmlenc-core/
15) Tolone, W., Ahn, G.-J., Pai, T. and Hong, S.-P.: Access control in collaborative systems, *ACM Computing Surveys*, Vol.37, No.1 (2005).
16) Bertino, E., Mella, G., Correndo, G. and Ferrari, E.: An infrastructure for managing secure update operations on XML data, *Proc. ACM Symposium on Access Control Models and Technologies* (2003).
17) Pan, C.-C., Mitra, P. and Liu, P.: Semantic access control for information interoperation, *Proc. ACM Symposium on Access Control Models and Technologies* (2006).
18) Jin, J. and Ahn, G.-J.: Role-based access management for ad-hoc collaborative sharing, *Proc. ACM Symposium on Access Control Models and Technologies* (2006).
19) Tan, W.C.: Research Problems in Data Provenance, *IEEE Data Engineering Bulletin*, Vol.27, No.4 (2004).
20) Buneman, P., Khanna, S. and Tan, W.C.: Why and Where: A Characterization of Data Provenance, *Proc. International Conference on Database Theory* (2001).
21) Buneman, P., Chapman, A. and Cheney, J.: Provenance management in curated databases, *Proc. 2006 ACM SIGMOD International Conference on Management of Data* (2006).
22) Buneman, P., Chapman, A., Cheney, J. and Vansummerenn, S.: A Provenance

Model for Manually Curated Data, *Proc. International Provenance and Annotation Workshop* (2006).
23) Tan, V., Groth, P., Miles, S., Jiang, S., Munroe, S., Tsasakou, S. and Moreau, L.: Security Issues in a SOA-based Provenance System, *Proc. International Provenance and Annotation Workshop* (2006).
24) Myers, A.C.: JFlow: Practical Mostly-Static Information Flow Control, *Proc. Symposium on Principles of Programming Languages* (1999).
25) Sabelfeld, A. and Myers, A.: Language-Based Information-Flow Security, *IEEE Journal on Selected Areas in Communications*, Vol.21, No.1 (2003).
26) 吉濱佐知子，工藤道治，小柳和子：動的アプローチによる言語ベースの情報フロー制御，情報処理学会論文誌，Vol.48, No.9 (2007).
27) Ferrari, E., Samarati, P., Bertino, E. and Jajodia, S.: Providing flexibility in information flow control for object oriented systems, *Proc. IEEE Symposium on Security and Privacy* (1997).
28) Chong, S. and Myers, A.C.: Security policies for downgrading, *Proc. ACM Conference on Computer and Communications Security* (2004).
29) Zdancewic, S. and Myers, A.: Robust declassification, *IEEE Computer Security Foundations Workshop* (2001).
30) Li, P. and Zdancewic, S.: Downgrading policies and relaxed noninterference, *Proc. ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (2005).
31) Bertino, E., Castano, S., Ferrari, E. and Mesiti, M.: Controlled access and dissemination of XML documents, *Proc. International Workshop on Web Information and Data Management*, (1999).
32) Damiani, E., di Vimercati, S.D.C., Paraboschi, S. and Samarati, P.: Securing XML Documents, *Proc. International Conference on Extending Database Technology* (2000).
33) Mazzoleni, P., Bertino, E., Crispo, B. and Sivasubramanian, S.: XACML policy integration algorithms, *Proc. ACM Symposium on Access Control Models and Technologies* (2006).
34) Chatvichienchai, S., Anutariya, C., Iwaihara, M., Wuwongse, V. and Kambayashi, Y.: Towards Integration of XML Document Access and Version Control, *Proc. Database and Expert Systems Applications* (2004).
35) Chatvichienchai, S. and Iwaihara, M.: Detecting Information Leakage in Updating XML Documents of Fine-Grained Access Control, *Proc. Database and Expert Systems Applications* (2006).
36) Iwaihara, M., Chatvichienchai, S., Anutariya, C. and Wuwongse, V.: Relevancy based access control of versioned XML documents, *Proc. 10th ACM Symposium on Access Control Models and Technologies* (2005).
37) Damiani, E., di Vimercati, S.D.C., Paraboschi, S. and Samarati, P.: A fine-grained access control system for XML documents, *ACM Trans. Information and System Security*, Vol.5, No.2 (2002).
38) 宮崎邦彦，洲崎誠一，岩村　充，松本　勉，佐々木良一，吉浦　裕：電子文書墨塗り問題，情報処理学会研究報告（CSEC）74 (2003).
39) 洲崎誠一，松本　勉：電子署名アリバイ実現機構—ヒステリシス署名と履歴交差，情報処理学会論文誌，Vol.43, No.8 (2002).
40) Rönnau, S., Scheffczyk, J. and Borghoff, U.M.: Towards XML version control of office documents, *Proc. ACM Symposium on Document Engineering* (2005).
41) Carminati, B. and Ferrari, E.: AC-XML documents: Improving the performance of a web access control module, *Proc. ACM Symposium on Access Control Models and Technologies* (2005).
42) Wang, Y., DeWitt, D.J. and Cai, J.-Y.: X-Diff: An Effective Change Detection Algorithm for XML documents, *Proc. 19th International Conference on Data Engineering* (2003).
43) Jaeger, T., Butler, K., King, D.H., Hallyn, S., Latten, J. and Zhang, X.: Leveraging IPsec for mandatory access control across systems, *Proc. 2nd International Conference on Security and Privacy in Communication Networks* (2006).
44) 宗藤誠治，中村めぐみ，工藤道治：プラットフォーム構成認証局，2007 年暗号と情報セキュリティシンポジウム (2007).
45) Katsuno, Y., Kudo, M., Watanabe, Y., Yoshihama, S., Perez, R., Sailer, R. and van Doorn, L.: Towards Multi Layer Trusted Virtual Domains, *Proc. 2nd Workshop on Advances in Trusted Computing* (2006).
46) Griffin, J.L., Jaeger, T., Perez, R., Sailer, R., van Doorn, L. and Cáceres, R.: Trusted Virtual Domains: Toward Secure Distributed Services, *1st Workshop on Hot Topics in System Dependability* (2005).

(平成 19 年 11 月 30 日受付)
(平成 20 年 6 月 3 日採録)

三品　拓也（正会員）

2004 年筑波大学大学院理工学研究科修了，修士（情報工学）．同年 4 月日本アイ・ビー・エム（株）入社，東京基礎研究所にてセキュア文書管理技術をはじめとする情報セキュリティ分野の研究に従事．

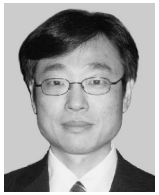


勝野　恭治（正会員）

1998 年慶應義塾大学大学院理工学研究科計算機科学専攻修士課程修了．同年日本アイ・ビー・エム株式会社入社．東京基礎研究所主任研究員．2008 年筑波大学大学院システム情報工学研究科リスク工学専攻後期博士課程入学．2003 年ソフトウェア科学会高橋奨励賞受賞．情報セキュリティ，コンピューター・ネットワーク，エージェント技術に関する研究開発に従事．日本ソフトウェア科学会会員．

吉濱佐知子（正会員）

1993 年青山学院大学経済学部経済学科卒業．同年（株）セック入社．2001 年より IBM T.J. Watson 研究所勤務，2003 年より現在まで日本アイ・ビー・エム（株）東京基礎研究所勤務．2007 年情報セキュリティ大学院大学情報セキュリティ研究科修士課程修了．トラステッド・コンピューティング，情報フロー制御，Web 2.0 アプリケーションセキュリティ等の情報セキュリティ分野の研究に従事．ACM 会員．



工藤　道治（正会員）

1988 年東京大学大学院工学系研究科修士課程修了，同年日本アイ・ビー・エム株式会社東京基礎研究所入社．情報セキュリティの研究・開発に携わる．主にアクセス制御・セキュリティポリシの研究に従事．2001 年に米国標準化団体 OASIS において XACML 委員会を設立，2003 年 2 月に国際標準になる．2002 年東京大学より博士（工学）の学位授与．現在，東京基礎研究所システム管理&コンプライアンス部門のシニアリサーチャー．電子情報通信学会会員．