**SCIS 2014** The 31st Symposium on
Cryptography and Information Security
Kagoshima, Japan, Jan. 21 - 24, 2014
The Institute of Electronics,
Information and Communication Engineers

# 機密情報の拡散追跡における追跡対象の削減
# Reducing Tracking Targets on Tracking Sensitive Information

前田 明彦* 高橋 健一 * 川村 尚生 * 菅原 一孔 *
Akihiko Maeta Kenichi Takahashi Takao Kawamura Kazunori Sugahara

**あらまし** 近年，情報漏洩事件が多発・深刻化している．個人情報漏洩の原因の約 7 割は,「管理ミス」,「誤操作」が占めている．このうち「誤操作」は，計算機利用者の不注意による操作ミスである．この対策として機密情報の拡散をカーネルレベルで追跡し，書き出しを確認することや制御することを目的としたシステムが提案されている．しかし，このシステムではプロセスに注目し，プロセスが機密情報を読み込んでいれば書き出した情報にも機密情報が含まれている可能性があると判断し，書き出された情報も追跡の対象となる．そのため，機密情報を読み込んだプロセスが情報を複数のファイル書き出すと追跡対象の情報が膨大な数になる．また，膨大な数の追跡対象の中にアプリケーションの設定ファイルやログファイルなどが含まれることが考えられる．それらのファイルが追跡対象となることも追跡対象の発散の原因である．本稿では，読み込まれた機密情報と書き出される情報を比較することで追跡対象の発散を抑える仕組みを実現する．

**キーワード** 情報漏洩対策, 誤操作, linux カーネル, システムコール, 機密情報の拡散追跡

## 1 はじめに

近年，計算機性能の向上や普及により今まで紙媒体で管理していた情報を計算機で管理する機会が増大している．また，情報を電子化することで，一度に大量の情報を持ち出すことも可能となり情報の提供や共有が容易になった．しかし，計算機で管理する情報は機密性の高い情報から機密性が低い情報までさまざまであり，機密性の高い情報が外部に漏れてしまうと企業や個人にとって大きな損失になる．個人情報漏洩のインシデント分析結果 [1] によると個人情報漏洩の原因の約 7 割は,「管理ミス」,「誤操作」が占めている．これらの情報漏洩は外部からの攻撃ではなく組織の内部のミスや誤操作が原因である.「管理ミス」とは，情報の公開・管理ルールの明確化や遵守が不十分であることが原因である.「誤操作」については，メールの送信先の宛先の間違いや操作ボタンの押し間違いなど，計算機利用者の不注意が原因である．計算機利用者の不注意をなくすことで誤操作による情報漏洩が防止できる．計算機利用者の不注意は，計算機操作の確認や制御によりある程度の防止ができる．そのため，本研究では，原因のひとつである「誤操作」に論点をおく．悪意のある計算機利用者や計算機利用者の意図した機密情報の持ち出しは対象としない．

「誤操作」を防止するための対策として，機密情報を追跡し，機密情報を送信または USB メモリなどの記憶媒体に書き出す際に警告を出すなどして操作を確認及び制御することを目的に，カーネルレベルで情報の拡散を追跡するシステム [2] が提案されている．しかし，プロセスに注目し，機密情報を読み込んだプロセスがファイルなどに情報を書き出した場合には，書き出したファイルも機密情報が含まれている可能性があり追跡の対象となる．そのため，機密情報を読み込んだプロセスが情報を書き出すごとに追跡対象が指数的に増加する．そこで，本論文では，追跡対象のファイルの絞り込む仕組みを実現する．

## 2 機密情報の拡散追跡システム

### 2.1 機密情報の拡散

機密情報とは個人情報などの機密性の高い情報が含まれた情報であり，本稿ではファイル単位で扱う．ファイルへの操作を記録することで機密情報の拡散を把握する．ファイルに含まれる機密情報の読み書きには，必ずメモリやハードウェアにアクセスするため，情報拡散の漏れをなくすためにメモリやハードウェアを直接コントロースするカーネルレベルの操作で把握する．図 1 に機密情報が拡散する過程を示す．情報の拡散には，必ずプロセ

* 鳥取大学大学院工学研究科情報エレクトロニクス専攻 〒 680-8550 鳥取県鳥取市湖山町南 4-101. Graduate School of Engineering, Tottori University, 4-101 Koyama-Minami, Tottori, 680-8550, Japan. s082050@ike.tottori-u.ac.jp

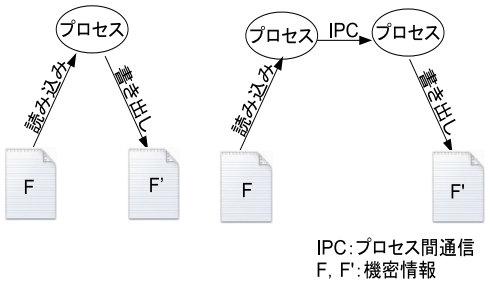


図 1: 機密情報の拡散

スが関係している．プロセスが情報を読み込み，読み込んだ情報をファイルへ書き出したり，プロセス間通信により情報が他のプロセスに伝搬し，拡散する．情報の拡散を追跡するためには，このような情報のプロセス間の伝達を把握する必要がある．

### 2.2 システムの概要

システムの概要を図 2 に示す．

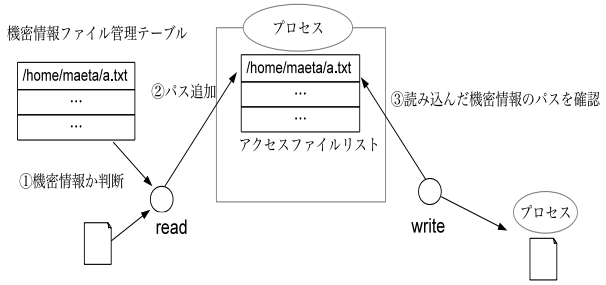


図 2: システムの概要

機密情報の拡散は，プロセスが機密情報を含むファイルを読み込み，プロセス間通信によりデータが拡散し，データがファイルに書き出されることで発生する．そこで，機密情報ファイル管理テーブルとアクセスファイルリストを準備する．機密情報ファイル管理テーブルでは機密情報を含む可能性のあるファイルを管理する．アクセスファイルリストはプロセスごとに準備され，プロセスが読み込んだ（機密情報を含む可能性のある）ファイルのリストを管理する．データが書き出された際，アクセスファイルリストにあるファイルのデータが書き出された可能性のあるデータとなる．また，プロセス通信（ソケット通信）発生時にデータの拡散を追跡するためには，そのデータに機密情報を含むかをデータを受信したプロセスが知る必要がある．そこで，送信するデータをアクセスファイルリストと共に送ることでプロセス間通信によるデータの拡散に対処する．

#### 2.2.1 機密情報ファイル管理テーブル

ファイル操作による拡散は，情報を読み込んだプロセスが，他のファイルに情報を書き出すことで情報の拡散が発生する．このため，ファイルを読み込んだ際に，読み込んだファイルが機密情報であるか判断する必要がある．そこで，機密情報と通常ファイルを区別するために，カーネル内に機密情報の絶対パスを管理するテーブルを用意する．ユーザは機密情報として扱いたいファイルを作成した場合，作成したファイルの絶対パスを登録する．ファイルを読み込むシステムコールである read システムコールが呼び出された際，読み込むファイルのパスが機密情報ファイル管理テーブルに含まれているか確認する．読み込むファイルが機密情報ファイル管理テーブルに含まれている場合，機密情報を読み込んだ可能性があると判断する．ここで，システムコールはカーネルモードで実行されるため，機密情報ファイル管理テーブルはカーネル内に準備する必要がある．このため，ユーザモードとカーネルモードで情報をやり取りする方法が必要となる．そのため，カーネル内の情報を表示したりカーネル内のパラメータ値の表示や変更が可能である proc ファイルシステムを利用する．proc ファイルシステムを利用することでユーザモードからカーネル内にあるテーブルの機密情報のパスを編集できるようになる．

#### 2.2.2 アクセスファイルリスト

機密情報はプロセスによって読み込まれ，それが他のファイルやプロセスに書き出されることで拡散する．書き出した情報が機密情報であった場合，書き出した先のファイルも機密情報として扱わなければ，情報漏洩につながる可能性がある．そこで，書き出す情報が機密情報であるか判断する必要がある．情報の書き出しは，write システムコール，プロセス間通信では sendto システムコール等により発生する．しかし，これらのシステムコールで書き出す情報はすでにファイルから読み込まれているため，書き出す情報が機密情報であるか判断することができない．

そのため，プロセスが機密情報を含むファイルを読み込んだ際に，このファイルのパスをアクセスファイルリストに登録する．プロセスが情報を書き出した場合に，アクセスファイルリストにパスが含まれていれば書き出した情報が機密情報を含む可能性があると判断する．Linux では実行する基本的な単位をプロセスとして管理しており，プロセスに関連するさまざまな情報を task_struct という構造体 (プロセスディスクリプタ) に格納している．プロセスディスクリプタは 1 つのプロセスに必ず 1 つ存在し，プロセスとプロセスディスクリプタは 1 対 1 で対応している．そのため，プロセスディスクリプタのメンバとしてアクセスファイルリストを追加する．

read システムコールで機密情報を読み込む毎にアクセスファイルリストへファイルのパスを追加する．プロセスは情報を書き出すときにアクセスファイルリストを

参照し，リストにパスがひとつでも格納されていれば，書き出された情報は機密情報の可能性がある．書き出した情報が機密情報の可能性がある場合には，書き出したファイルのパスを機密情報ファイル管理テーブルに追加する．これによって書き出されたファイルも機密情報として扱われ，ファイルの読み書きによる機密情報の追跡が可能となる．

#### 2.2.3 プロセス間通信

プロセス間通信では，プロセス同士で通信しているデータが機密情報であるかどうかをカーネルからでは判断できない．そこで，機密情報を受信した際に受信した情報が機密情報であるか判断する必要がある．受信側では送られてきた情報が機密情報であることを認識するためには，送信側から受信側への送信した情報が機密情報であることを伝える必要がある．受信データが機密情報であるか判断させるため，送信するデータに加えてアクセスファイルリストのデータを送信する．これにより受信した側でも，送信されたデータが送信元のどのファイルのデータを含む可能性があるか知ることができる．機密情報を送信する場合のデータ構造を図 3 に示す．


図 3: 機密情報を送信する場合のデータ構造

送信するデータに機密情報が含まれている可能性がある場合，受信側に機密情報であると伝えるためにアクセスファイルリストに格納されているパスを本来送信するデータに連結する．また，パスと本来送信するデータを区別するための情報として，先頭に”PATH_SIZE”というタグをつけ，その後ろに本来送信するデータに連結したパスのサイズを追加する．受信側では，受信したデータの先頭が”PATH_SIZE”であれば，受信したデータに機密情報が含まれていると判断する．そして，”PATH_SIZE”の後ろに連結されているサイズにより，アクセスファイルリストと本来受信するデータを分割する．このことで，送信側から受信側へ機密情報を送信したことを伝える．

## 3 追跡対象の削減

### 3.1 既存システムの問題点

2.2 節で述べた機密情報の拡散追跡システムにより機密情報の拡散は追跡可能である．機密情報の拡散追跡システムでは，プロセスに注目し，機密情報を読み込んだ時にアクセスファイルリストにパスを追加し，アクセス

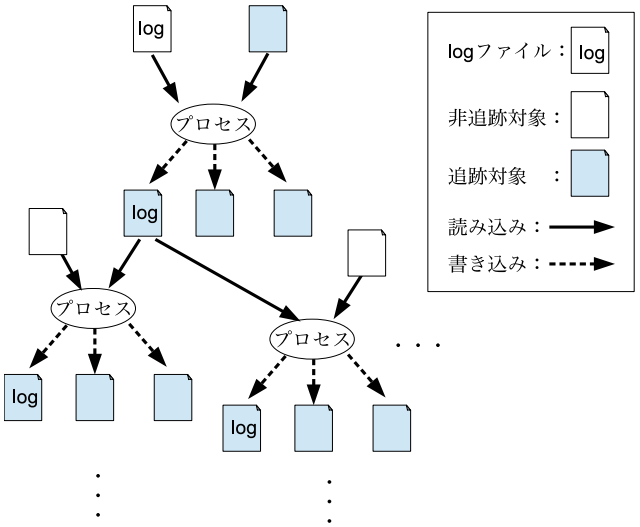


図 4: 追跡対象が発散するイメージ

ファイルリストを参照することでプロセスが機密情報を読み込んだことが分かる．そして，情報を書き出す時にはアクセスファイルリストを参照し，パスが格納されていれば，書き出した情報も機密情報として登録する．

この方法では，書き出した情報を機密情報とするかは，書き出した情報に関わらず，情報を書き出すプロセスが機密情報を読み込んでいるかどうかで判断する．しかし，アプリケーションを利用する場合，プロセスはアプリケーションの設定ファイルやログファイルなど様々なファイルを読み書きすることが考えられる．このため，機密情報を読み込んだプロセスがファイルにデータを書き出す場合，機密情報とは関係ないデータ，例えば，ログをログファイルに書き込む場合にはログファイルが機密情報として機密情報ファイル管理テーブルに登録される (図 4)．ログファイルが機密情報ファイル管理テーブルに登録されると，そのログファイルを読み込むすべてのプロセスが書き出すデータはすべて機密情報として扱われ，追跡対象が膨大な数になることが予想できる．

そこで，関連研究 [2] では設定ファイルなどを計算機利用者が登録し，追跡対象から除外する機能を実装している．しかし，それらはあらかじめ除外しておかなければならない．また，計算機利用者が判断するのは手間がかかる．その対処として，設定ファイルの候補を通知する仕組みを実装しているがこれも利便性の低下につながる．そこで機密情報を読み込んでいるプロセスが機密情報を書き出した可能性があるということだけで書き出した情報を追跡対象とするのではなく，書き出した情報が本当に機密情報であるか特定する必要がある．

### 3.2 問題への対処

3.1 節で述べた問題への対処方法として，データフローを考慮した研究 [3] やコンパイラによりプロセス内のデータの流れを把握する研究 [4] がある．これらの研究では対象のファイルにデータ提供者の意図する保護方針を定義したデータ保護ポリシやラベルをつけることでデータの流れを特定している．そのため，計算機利用者が保護したいファイルなどにラベルなどを設定する必要がある．しかし，ファイルに保護ポリシやラベルをつける手間がかかる．このため，ファイルの指定だけで拡散が追跡でき，書き出された情報が機密情報であるか特定することが望ましい．

そこで，本研究では，読み込まれた機密情報と書き出す機密情報の比較することで，追跡対象の絞り込みを行う．これにより，追跡対象の削減が実現できると考える．また，本研究では計算機利用者の誤操作を対象としており，ファイル単位のコピー&ペースとソケット通信でファイルを送信することを対象とする．このため，プロセスが読み込んだデータを編集することは対象としない．

### 3.3 比較方法の検討

読み込まれた機密情報と書き出される情報の比較を行うためには，どの情報を比較するかが重要である．比較方法として，4 つの方法を検討した．

**・読み込んだ情報と書き出す情報のバイト数を比較**

ファイル全体のコピー&ペーストやファイルの送信を対象とし，読み込んだ機密情報のバイト数と書き出す情報のバイト数を比較することで機密情報の拡散を追跡できる．この方法のメリットはデータの長さを比較するためデータ自体を比較することなく処理が軽い．一方，デメリットは，バイト数で比較しているため 1 バイトでも増減していると追跡ができなくなる．そのため，読み込まれた情報の中と書き出す情報の中を比較する方法が必要であると考えた．

**・それぞれの情報のハッシュ値を比較**

読み込んだ機密情報と書き出す情報のハッシュ値をとり，比較する．ハッシュ値は同じデータからは同じハッシュ値しか求められない．このため，ハッシュ値を比較することで一致すれば必ず機密情報であると判断できる．バイト数の比較では中身のデータを比較していないため，偶然同じバイト数であれば追跡対象とされる．そのため，バイト数で比較する方法よりは信頼性が高いが，ハッシュ値の計算コストがかかる．

**・先頭 n バイトを比較**

読み込まれた情報と書き出される情報の先頭 n バイトを比較する．ファイルのコピー&ペーストやファイル送信を行う場合は，プロセスによって読み込んだデータを編集することなく書き出されるため，先頭の n バイトの比較をすることで，読み込まれた機密情報と書き出された情報が一致しているか判断できる．しかし，ファイルの先頭にヘッダーなどの決まった情報が書かれている場合には，読み込んだ情報と書き出した情報が一致してしまう．このため，ファイルの先頭が同じであるだけで機密情報でない情報が追跡対象となってしまうことが考えられる．

**・終端の n バイトを比較**

読み込まれた情報と書き出される情報の終端 n バイトを比較する．終端の n バイトを比較することでヘッダーなどを考慮することなく，読み込んだ機密情報と書き出す情報の比較を行うことができる．

## 4 実装および評価

### 4.1 実装

先頭 n バイトを比較する方法を Linux(3.9.6) 上に実装した．プロセッサは Intel(R)Core(TM)i7-3770K CPU @ 3.50GHz × 4，メモリは 8GB を搭載した PC で評価した．n は 10 とし，読み込まれた機密情報の先頭 10 バイトと書き出される情報の先頭 10 バイトを比較する．読み込まれた機密情報の先頭 10 バイトは，read システムコールでパスを取得するのと同時に読み込んだ情報の先頭 10 バイトも取得する．読み込んだ機密情報の先頭 10 バイトの保持は，2.2.2 節のアクセスファイルリストを拡張し，アクセスファイルリスト内で管理しているパスと組にして管理する．

| 読み込んだ機密情報のパス | 先頭 10 バイト |
|---|---|
| /home/maeta/secret1.txt | Address:To |
| /home/maeta/secret2.txt | Tel:XXX-YY |

図 5: アクセスファイルリストの拡張

書き出される情報との比較は，write，sendto システムコール等の中で書き出される情報の先頭 10 バイトを取得し，アクセスファイルリストに格納されている 10 バイトと比較する．アクセスファイルリストにはパスと先頭 10 バイトを組にして格納しているため，一致すれば，書き出す情報が機密情報であることが分かる．ファイルに書き出した場合は書き出した追跡対象とする．送信する場合は 2.2.3 節で述べた方法で送信し，受信側へ機密情報であることを伝える．

### 4.2 実験

読み込みと書き込みの情報を比較しない場合と比較する場合で追跡対象が削減できるか確認する．

実験では，機密情報を 1 ファイル，非機密情報を 1 ファイル用意した．同時に 2 つのファイルをアプリケーション (Libre Office 4.1, emacs 24.3) で開き，機密情報の内容をすべて非機密情報のファイルにコピーする．データ保存後，追跡対象となっている数を比較する．実験結果を表 1 に示す．

表 1: 追跡対象の削減実験の結果

| アプリケーション | 比較なし | 比較あり |
|---|---|---|
| libre office | 9 | 2 |
| emacs | 4 | 2 |

理想の追跡対象の数は，もともと機密情報であったファイルと機密情報がコピーされたファイルの 2 つである．比較なしの場合は，Libre office では，この 2 ファイル以外に gtk の設定ファイルや Libre Office の設定ファイル，Libre Office のログファイルなど 7 つのファイルが追跡対象となっていた．これは，機密情報を読み込んだプロセスが書き出す情報に関係なく，書き出したファイルを追跡対象として登録しているからである．emacs の場合でも，理想の追跡対象の数より 2 ファイル多く登録されていた．一方，今回実装した比較ありの場合は，理想通り追跡対象は 2 ファイルとなった．これは，読み込んだ機密情報と書き出す情報を比較しているため，設定ファイルやログファイルなどのアプリケーションに関するファイルへの書き出す情報と機密情報が区別できているからである．実験では，1 ファイルにしか情報を書き出していないが，複数のファイルに書き出した場合は追跡対象の発散が予想できる．しかし，読み込んだ機密情報と書き出す情報の比較を行うことで設定ファイルやログファイルなどを追跡対象として登録することがなくなり，追跡対象を削減することができた．

### 4.3 今後の課題

本研究では，読み込んだ機密情報と書き出す情報の比較を行うために，情報の先頭 10 バイトを比較した．先頭 10 バイトの比較では，情報を読み込んだアプリケーションで先頭の 10 バイトが編集されてしまった場合，書き出したファイルは追跡対象として登録されないため，これ以降の追跡ができなくなる．また，ファイルの先頭にヘッダーなど決まったフォーマットで書かれている場合は，先頭 10 バイトが同じ可能性があり，機密情報とそうでない情報の区別ができなくなる．

情報を書き出す場合に，読み込んだ機密情報と書き出した情報を比較することにより，追跡対象の削減効果があることは分かったが，どの情報を比較するかやどのようにして比較すると追跡対象が漏れなく登録できるか検討する必要がある．また，提案システムの処理性能を評価するため，システムコール実行時間のオーバーヘッドを評価する必要がある．

## 5 おわりに

本論文では，追跡対象のファイルを絞り込むことを目的とし，読み込まれた機密情報と書き出された情報の比較を行うことで追跡対象のファイルの削減を実現した．しかし，今回は，読み込まれた情報と書き出された情報の先頭 10 バイトを比較したため，先頭の情報が編集されると追跡ができなくなる．また，ヘッダーなどが付与されている場合は，先頭の情報の比較では不十分である．今後の課題は，読み込まれた情報と書き出された情報の比較方法を検討すると共に比較によるオーバーヘッドを評価することである．

## 謝辞

本研究は科学研究費補助金（23700027，23700098）の支援を受けて行なった．

## 参考文献

[1] NPO 日本ネットワークセキュリティ協会， ２０１０年情報セキュリティインシデントに関する調査報告書～個人情報漏洩編～， http://www．jnsa.org/result/incident/2010．html

[2] 田端 利宏，箱守 聰，大橋 慶，植村 晋一郎，横山 和俊，谷口 秀夫，“機密情報の拡散追跡機能による情報漏えいの防止機構”，情報処理学会論文誌，Vol. 50，No．9，pp．2088-2102，(9， 2009)．

[3] 井田 章三，河島裕亮，樫山武浩，瀧本栄二，毛利公一，“データフローを主体としたアクセス制御を実現する DF-Salvia の設計と開発”，情報処理学会第 73 回全国大会講演論文集，Vol．2011，No．1，pp．511-513，(3， 2011)．

[4] 齋藤 彰一，“プロセスの通信制限と公開鍵暗号によるファイルの移動制御方式の提案”，情報処理学会研究報告. [システムソフトウェアとオペレーティング・システム]，vol．2006，NO．86，pp．79-86．