デ-13

プライバシ保護を実現するコンテキストアウェアOS

鈴来 和久† 一柳 淑美† 西村 和憲†† 毛利 公一††† 大久保 英嗣†††
† 立命館大学大学院理工学研究科 †† 立命館大学理工学部 ††† 立命館大学情報理工学部

1 研究背景と目的

現在, ソフトウェアにおけるセキュリティ対策が重要な課題となっている. 特に, プライバシ情報, 著作権で保護されたコンテンツ, 機密情報などの重要なデータの漏洩を防ぐための技術開発が盛んに行われている.

従来のデータアクセス制御方式では, 読出し, 書込み, 実行の各アクセス要求に対して, アクセス権限の正当性を検査し, 実行の可否を制御する. 従来方式では, アクセス権限が正しく設定されている場合でも, 以下に示すようなデータ漏洩が発生する危険性が高い.

- 攻撃者によるアクセス権限の不正な取得や改変によるデータ漏洩
- 正当なアクセス権限を持つユーザの誤操作によるデータ漏洩
- 正当なアクセス権限を持ち, かつ不正行為を意図したユーザによるデータ漏洩

そこで, 我々は, データへのアクセス要求が発生した際に, ユーザ, 計算機, プロセスの状態 (コンテキスト) に着目し, コンテキストに応じたアクセス制御によりデータ漏洩を防止するオペレーティングシステム (以下, OS と記す) [1, 2] を開発している. 本 OS は, データ漏洩の原因となるプロセスの動作を制御し, プライバシ保護を実現する. また, OS でデータ保護を実現するため, プライバシ保護を考慮していないアプリケーションによるデータ漏洩を防ぐ.

2 コンテキストに適応したデータ保護

本 OS は, システムコールが発行された際に表 1 に示す情報を取得し, プロセスごとに動作履歴としてそれらを時系列で保存する. 取得した情報は, 読出し・書込み・通信といったデータ漏洩の原因となるシステムコールが発行された際に, 本 OS に実装したデータ保護機構から参照される. 特に, システムコールの実行可否の判定に用いる履歴情報を, 本稿ではコンテキストと定義する.

ユーザは, それらのコンテキストで構成されるデータ保護ポリシを定義できる. データ保護ポリシファイルが存在する場合, システムコール処理の前にデータ保護ポリシファイルを読み出す. 本 OS は, データ保護ポリシとプロセスの動作履歴に基づき, システムコールの実行を制御するフィードバックシステムモデル [2] を採用している. 例えば, ディスクに保存されたファイルに対するアクセスを制御する場合, 本 OS では `open`, `read`, `write` システムコールの実行を, コンテキストに適応して制御する. `open` システムコールが発行された際に取得するコンテキストは, 次の通りである.

表 1 システムコール発行時に取得する情報

取得元	項目
OS 内部	実ユーザ ID, 実効ユーザ ID, プロセス ID, 相対時刻
アプリケーション	システムコール番号, 引数, 返り値
ハードウェア	現在時刻, 無線 LAN アクセスポイントの ESSID, 電波強度

プロセス ID 制御対象のプロセスか否か検査する.

(実効) ユーザ ID システムコールの実行が許可されているユーザか否か検査する.

システムコールが発行された時刻 当該プロセスに対して時間制約を課す場合に利用する.

アクセス対象のパス名 アクセスが許可されている資源か否か検査する. また, データ保護ポリシファイルが存在するか否か検査するために利用する.

フラグ 読出し, 書込み, 新規作成など, 当該プロセスの次の挙動を制御する際に利用する.

ファイルディスクリプタ `read` や `write` を制御する際に, アクセス対象の資源の特定に利用する.

3 コンテキストアウェア OS の全体構成

図 1 に, 現在実装しているコンテキストに適応したデータ保護を実現する OS の構成を示す. 我々が提案しているデータ保護手法が, 既存のアプリケーションに対して適用可能であることを示すために, 本 OS は, Linux カーネル 2.6.6 に変更を加える形で実装している. Context Watcher, Action Control Mechanism は, データ保護機能を実現するために実装したモジュールである. また, Context List は, 取得したコンテキストを時系列データとして管理するために, 本 OS に追加したデータ構造である. 実装方法の詳細, および処理手順は, 文献 [1, 2] で示している. また, 位置を取得するため, ESSID と電波強度を取得するためのモジュールを実装している.

4 デモンストレーションの概要

4.1 プロセスの動作履歴をコンテキストとするファイルアクセス制御

2 章で述べたように, 本 OS は, プロセスの動作履歴を把握することができる. このため, 資源へのアクセス状況によってプロセスに課す制限を変化させること, すな

†Graduate School of Science and Engineering, Ritsumeikan Univ.
††Faculty of Science and Engineering, Ritsumeikan Univ.
†††College of Information Science and Engineering, Ritsumeikan Univ.

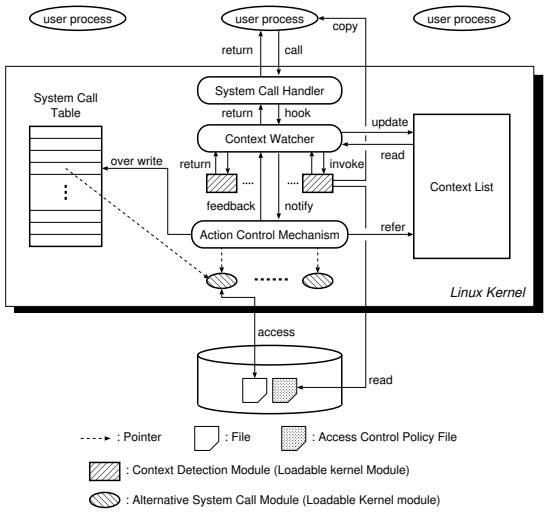

図 1　コンテキストアウェア OS の構成

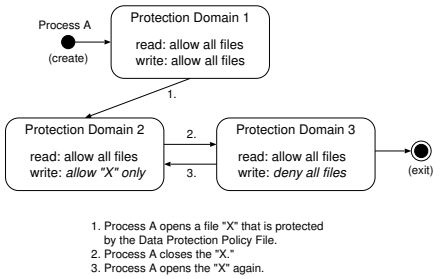

図 2　プロテクションドメインの動的遷移

わち, プロテクションドメインの動的遷移が実現できる (図 2 参照). そこで, プロセスのコンテキストに適応したファイルアクセス制御として, `open` システムコールから取得したコンテキストに適応して `read` と `write` システムコールの実行を制御するデモンストレーションを実施する.

4.2　位置をコンテキストとするファイルアクセス制御

計算機やユーザに共通のコンテキストとして, これらの現在位置が挙げられる. 位置を取得する手法として, IEEE802.11 の無線 LAN で利用されている ESSID (Extended Service Set Identifier) と電波強度を利用する (図 3 参照). 電波強度が閾値を下回った場合, データ保護ポリシに基づきシステムコールの実行を制御する. これにより, アクセスポイントからの距離に適応したデータアクセス制御を実現する. そこで, 前節で述べたデモンストレーションに, 計算機の位置をコンテキストとして加えた場合のファイルアクセス制御に関するデモンストレーションを実施する.

プロセスの動作履歴や位置をコンテキストとして扱い, コンテキストに適応してファイルアクセスを制御することにより, ユーザにとって意味のある時間帯ごとのデータ保護と, 計算機の物理的な位置によるデータ保護が可能になる. 例えば, 学校において生徒が携帯端末を利用している場合, 授業中に教室内ではその授業のレジュメを見ることができるが, 授業中に教室以外でレジュメを見ることや, 授業以外の時間帯にレジュメを見ることを禁止することができる. また, プロセスの動作履歴に適応することにより, 不正コピーを防止できる. これにより, 顧客情報などの重要なデータを保存したファイルにアクセスしたプロセスは, 当該ファイル以外のファイルにデータを書き出すことを禁止することができる.

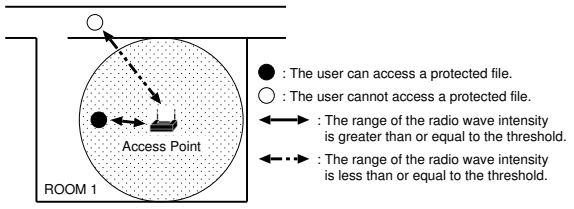

図 3　無線 LAN を利用した位置の検出

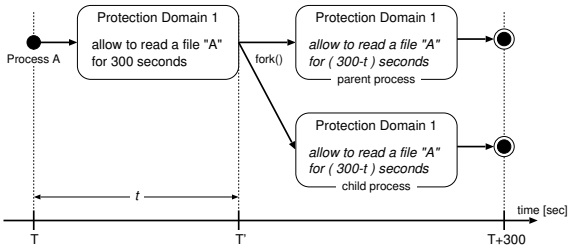

図 4　コンテキストの適応的継承

4.3　子プロセスの生成によるアクセス制御の継承

子プロセスを生成するプログラムにおいて, 親プロセスに課されているアクセス制限が子プロセスに課されない場合, 悪意のあるプログラムが, アクセス制限のない子プロセスを利用してデータを不正に流出させる危険性がある. このため, 本 OS では, 子プロセスを生成する際に, 親プロセスのコンテキスト (動作履歴) と, 親プロセスに課されているアクセス制限を子プロセスに継承させる. 本デモンストレーションでは, 時限つきファイルアクセスを例としたアクセス制御の継承を示す. 子プロセスは, 親プロセスに課されたアクセス制御を継承するため, 子プロセスがファイルにアクセスできる時間は, データ保護ポリシに記述された時限ではなく, 親プロセスがアクセスできる残り時間と等しくなる. これにより, コンテキストに適応したデータ保護を, 子プロセスを生成するプログラムにも適用できる.

参考文献

[1] 一柳 淑美, 鈴来 和久, 毛利 公一, 大久保 英嗣: コンテキストに適応可能なデータ保護機構におけるファイルアクセス制御, 情報処理学会第 66 回全国大会講演論文集 (1), pp. 95–96 (2004).

[2] 鈴来 和久, 一柳 淑美, 毛利 公一, 大久保 英嗣: プライバシ保護を実現するオペレーティングシステムにおけるコンテキスト管理手法, 情報処理学会研究会報告 2004-OS-96, Vol. 2004, No. 63, pp. 7–14 (2004).