TOMOYO Linux

タスク構造体の拡張によるセキュリティ強化Linux

[当日説明用資料]

平成16年6月3日
株式会社NTTデータ
技術開発本部
オープンシステムアーキテクチャグループ
原田季栄
haradats@nttdata.co.jp

Linux セキュリティ上の課題

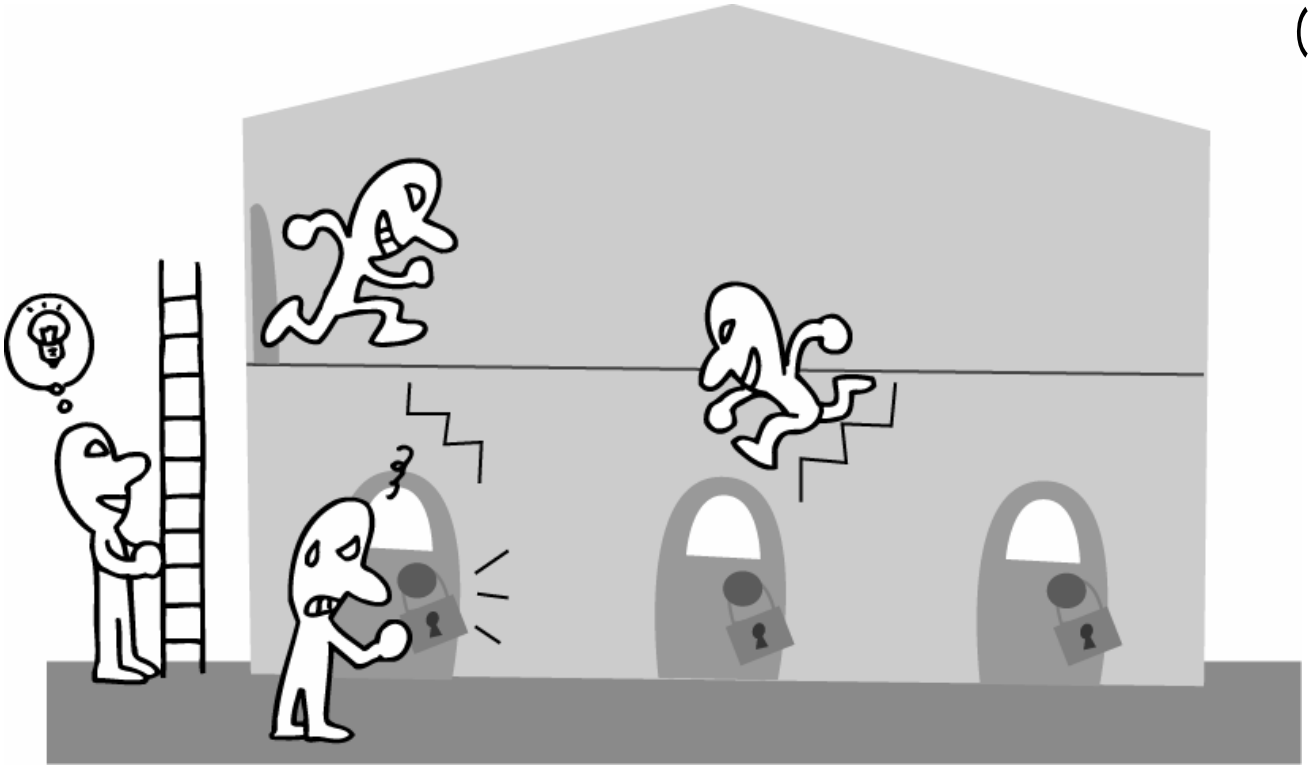
「所有者」による「自由裁量」のアクセス制御

粗い分類と、制御の粒度

システム管理者に対しては効力を持たない

システム管理者権限を奪われると歯止め無し

（文献 [1] ）



強制アクセス制御（MAC）

- Mandatory Access Control の略
- DoDが1985年に発令したTCSECの中で、”CLASS (B1): LABELED SECURITY PROTECTION” として記述
  - アクセスの主体（要求する側）と客体（要求された側）について
  - 付与された重要性 (sensitivity) のラベルに基づき
  - 「強制的に例外なく」可否を判定する

SELinux

- Security-Enhanced Linux
  - http://www.nsa.gov/selinux/
- **NSAが開発、公開しているセキュリティ強化Linux**
- **特徴**
  - **強制アクセス制御をLinuxに実装**
  - DTE (Domain Type Enforcement)
    - **プログラムの実行状況を「ドメイン」という概念で表現**
    - **「ドメイン」毎にアクセス許可を定義し、アクセスを制限する**
  - RBAC (Role Based Access Control)
    - **同一ユーザでも役割により異なる権限を与える**
  - LSM (Linux Security Modules) **に対応、2.6カーネルに組み込み済み**
  - **きめこまかなアクセス制御を実現できる**

SELinuxの運用

- 「ドメイン」を定義し、
- 「ドメイン」間の遷移を定義(execを唯一の契機として)し、
  - ドメインの粒度は管理者まかせだが、もっとも細かい場合は全ての
    プログラムの起動でドメインを切り替えること＝現実には定義不可能
- 「ドメイン」毎のアクセス許可条件を記述する。
  - アクセス許可の粒度は「システムコール」(固定)
- (必要に応じて)RBACを記述する。
- 上記全ては管理者が定める「ラベル」に基づいて行われる。
  - ラベル付けの正しさは全ての前提

SELinuxで問題解決?

- 「適切なポリシー」の策定と運用が現実には困難
  - 実装と一緒に配布されているポリシーを修正して利用するのが一般的と思われる
  - permissiveモードで運用して、「不足しているアクセス許可」を抽出して、ポリシーに追加するのは容易だが・・・
    - 「過剰なアクセス許可」は検知しがたく、つけいられる隙となり得る
    - 「動くようになった」で良いのか?(良くないのでは?)

- ラベルによるアクセス制御の問題
  - 管理、運用上の負担
  - 定義も解析も確認もラベルを介して行うという不便さ

SELinuxの運用

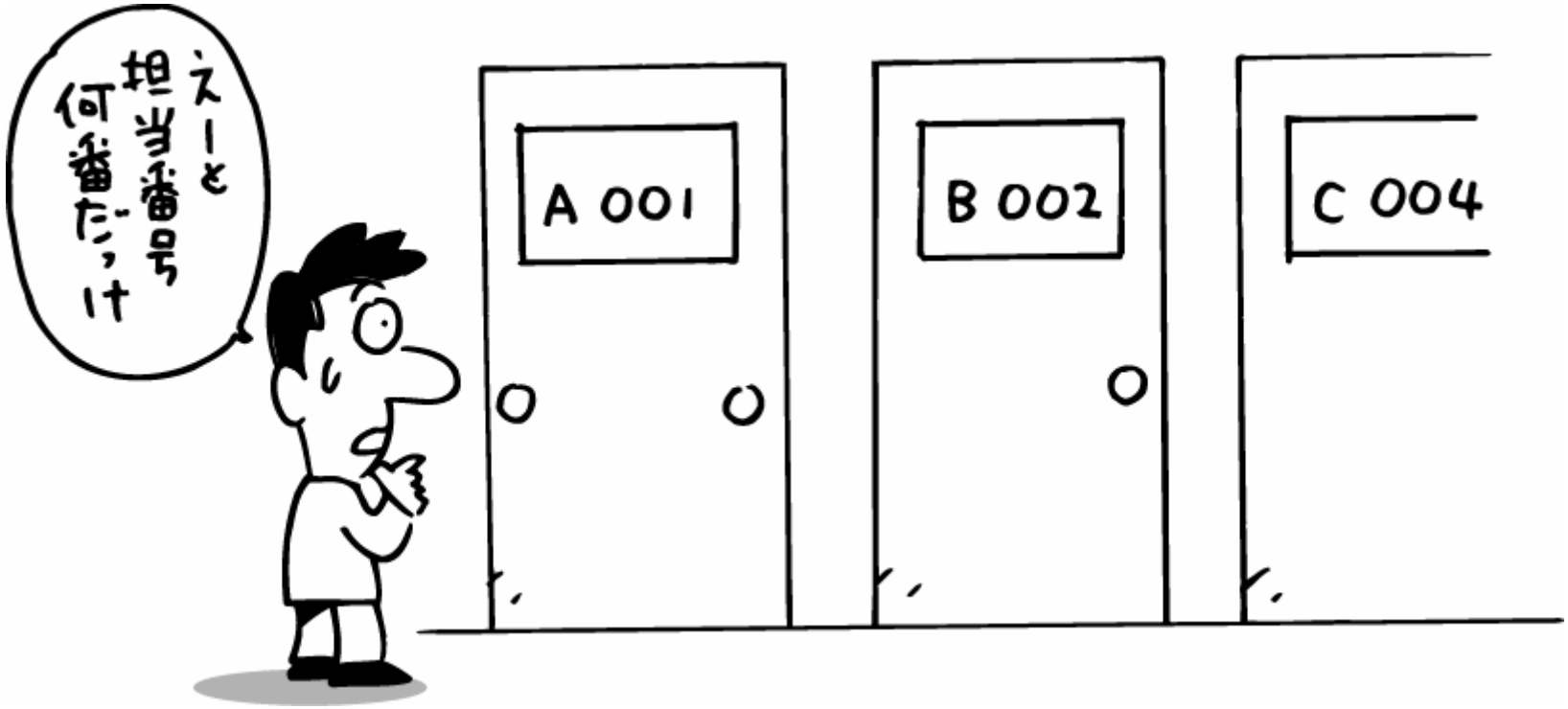
エラーログから必要なアクセス許可を割り出して
ポリシーに追加すれば「動作する」ように
することは可能だが・・・

ポリシーを定義するということは

- ポリシーはデフォルトで全てのアクセスを禁止するようになっている。
- 管理者はアプリケーションが動作するために必要なものを全て正確に書き出さなければならない。

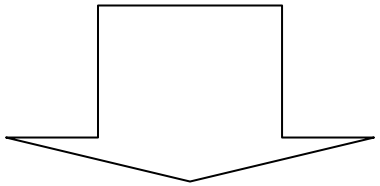
ラベルによるアクセス判定

ユーザが意識するファイル名やディレクトリ名ではなく、それらにつけられた符号（ラベル）によりアクセス可否が判断される。



TOMOYO以前の取り組み

- **アクセスポリシー自動生成支援システム**
  - JNSA Network Security Forum 2003**で発表**
    - http://www.jnsa.org/award/2003/result.html
  - **特殊なカーネルにより、**
    - **プロセス起動履歴と、履歴毎のアクセス要求を記憶**
    - **記憶したアクセス要求をファイルに抽出**
    - **Windows上でGUIエディタを用いて確認、編集**
  - **特徴**
    - **もれがなく、正確なポリシーの策定が可能**



これに強制アクセス制御機能を加えたものが TOMOYO

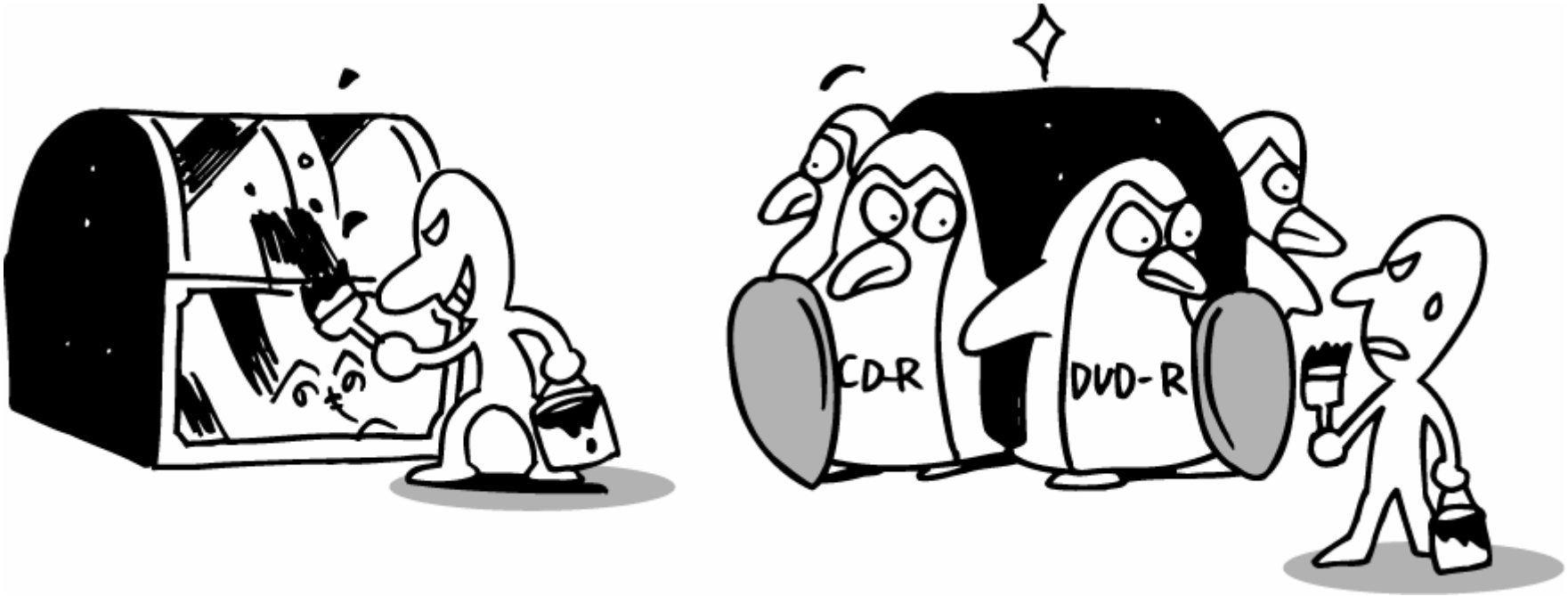
TOMOYO以前の取り組み

SAKURA　Linux（物理的な改ざん防止、LC2003で発表）

root fsがread onlyで動作するLinuxを実現

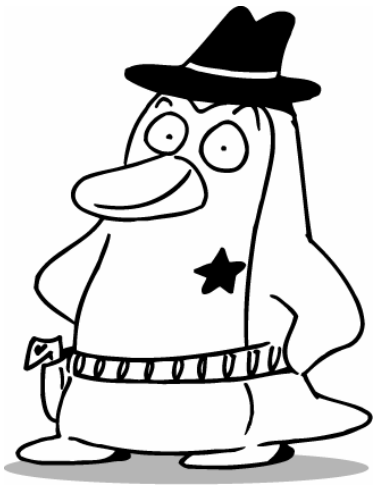
管理者権限を奪われても絶対に改ざんできない

物理的なので確実、ポリシー不要



SAKURAの限界

- 改ざん防止策にはなるが、情報漏洩防止策にはならない
  - 改ざんはできないが、アクセスを制御していないから、管理者権限を奪われると参照は可能となってしまう
- かといって複雑なポリシーの管理運用はしたくない
- そこで開発されたのがTOMOYOです



TOMOYO Linux

- ラベルによらないアクセス許可の付与
  - ファイル名、ディレクトリ名そのままでポリシーを確認、編集
- アクセスポリシーの自動定義機能

- アクセスポリシーに基づく強制アクセス制御
- アクセスポリシーによらない自発的アクセス制御

（詳細については論文を参照ください）

ラベルによらないアクセス許可

普段意識しているファイル名やディレクトリ名を
そのまま使えるから直感的でわかりやすく、
間違えない

ポリシーの自動定義

管理者の手をわずらわせることなく
必要十分なポリシーが得られる

SAKURAとTOMOYO

「改ざん防止のSAKURA」+「アクセス制御のTOMOYO」で
簡単ながら強固なセキュリティを実現可能

タスク構造体について

- 今回の強制アクセス制御の実装ではタスク構造体を利用しています。
  - 例外なく全てのプロセスが持っている
  - `fork()`, `exec()`(「複製」と「更新」)により引き継がれていく
  - 詳細については、解説用資料および論文を参照ください

デモ1

- アクセスポリシー自動定義
  - 自動定義モード(ACCEPT)カーネルで起動
  - 発生したアクセス許可条件が、自動的に記録される
    - プロセスの起動履歴毎に（同じmountプログラムであったとしても、その/sbin/initからの系統図により区別される。SELinuxで言えば、「全てのプロセスの起動履歴を異なるドメインに対応づけた」状態＝事実上のもっとも細かい粒度）
    - 記録されるのは、プロセスとアクセス対象とアクセスモード
  - ポリシーはテキストエディタで確認、編集可能
- 自動定義された内容に基づく強制アクセス制御
  - 強制モード(ENFORCE)カーネルで起動すると、定義された内容以外はアクセスが失敗する

デモ2

- アクセスポリシーによらない自発的アクセス制御
  - 通常のLinuxでは、setuid rootされた実行可能ファイルは、誰が実行しても常にroot権限で動作します。
  - そのようなプログラムが乗っ取られてしまうと、大変危険な事態となります。
  - そこで、「必要がなくなったらroot権限を得る権利を自分で放棄できる」よう実装しました。

デモ3

- サーバー機能
  - SAKURAとTOMOYOの組み合わせ
    - 「改ざん防止」+「独自強制アクセス制御」
  - USBフラッシュメモリから起動
  - Apache等サーバをWindowsXP端末からアクセス
  - この状態で改ざん防止と、アクセス制御が有効となっています

LinuxWorld Expo展示

- 本日．org Pavilionにて、TOMOYO Linuxのデモを行っています。
- お気軽にお立寄りください。

公開、質問について

- 開発が一段落したら、是非公開したいと考えています。
- 状況については、「参考文献」のサポートページで更新します。
- 説明および資料について質問があれば、タイトルページのメールアドレス宛ご連絡ください。

参考文献

[1] 日経システム構築 2004年4月号 no.132「解説」

– 「セキュアなシステムを作る(3つの原則に従いOSの機能を強化)」

[2] 読み込み専用メディア上でのLinuxサーバの運用について

– 「読み込み専用マウントによる改ざん防止Linuxサーバの構築」

Linux Conference 2003

http://lc.linux.or.jp/lc2003/30.html

原田季栄、保理江高志、田中一男

[3] 強制アクセス制御のポリシー定義の自動化について

– 「プロセス実行履歴に基づくアクセスポリシー自動生成システム」

Network Security Forum 2003

http://www.jnsa.org/award/2003/result.html

原田季栄、保理江高志、田中一男

[4] サポートページ

-http://www11.plala.or.jp/tsh/

– 本資料掲載内容のフォローアップ情報を公開します