Linux CONFERENCE

Linux Conference 2003

「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」

平成15年10月30日
株式会社NTTデータ
技術開発本部
原田季栄 haradats@nttdata.co.jp

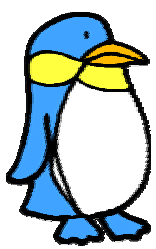
発表内容

- 読み込み専用マウントによる改ざん対策
  - CD-R等を利用した物理的な書き込み保護
- 改ざん対策をより強固にするための修正
  - マウント等の制限によるシステムの保護
  - シェルコード対策
- 既存技術を使用したセキュリティ向上策
- 構築したプロトタイプシステムの紹介

目次

開発の経緯

- ポリシーベースのセキュリティ強化Linux (SELinux、SubDomain等) を使用してみた。
  - インストールは特に問題なし
  - アプリケーションの互換性は保たれている
  - 非常に高い粒度でアクセス制限を記述可能
  - システム管理者も制限を回避できないので有効
- 感想
  - ポリシーによる強制的なアクセス制御はあくまで手段
  - 「適切なポリシー」の管理運用 (特に導入時) が課題となる
  - セキュリティ改善策として有効だが万人向きとは思えない
- ポリシーの管理運用なしにできることは無いか？

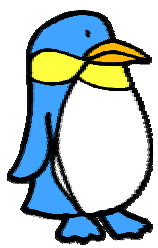


目標の設定とアプローチ

- 目標設定
  - ソフトウェアやドキュメントの公開等、頻繁に更新されない情報を扱うwebサーバを
  - 管理者にポリシーの管理運用という新たな作業負担を伴うことなく
  - 安全、確実に改ざんから守る
- アプローチ
  - 読み込み専用マウントを基盤として
  - Linuxが標準で備える機能を活用しながら
  - 最小限の修正で

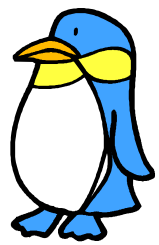
読み込み専用化の課題

- Linuxは root fs（/ ディレクトリにマウントするパーティション）を読み込み専用にしても動作するか？
  - /devへは書き込みが発生するが、Linuxでは/devはroot fsと同一パーティションになければならない。
  - 回避策
    1. devfs の利用
    2. tmpfs と mount --bind オプションの利用

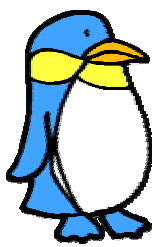


読み込み専用化の課題

- システムのログやアプリケーションの動作には読み書き可能領域が必要。
  - 具体的には、/var や/tmp
  - 対処方法
    - 必要に応じてハードディスクをマウント
    - 揮発しても良い内容は tmpfs を使用

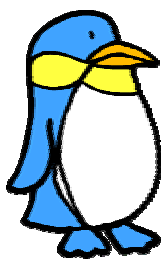


- root fs を読み込み専用モードでマウントしても、システム管理者権限を奪われると/dev/* 経由での改ざん、破壊が可能。
  - root fsを物理的に書き換えができないメディアに保持する。



読み込み専用化の課題

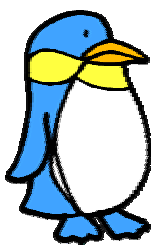
- root fs が物理的に保護されたメディアに置かれていれば安全か？
  - システム管理者権限があれば任意のディレクトリに対して (再) マウントが可能。
    - root fsの一部または全部を隠蔽される・・・
    - バックドアが仕込まれた書き込み可能領域をroot fsの上にマウントされる・・・



マウントに制限を加える。

読み込み専用化の課題

- カーネルにマウント等を制限する機能を追加
  - マウントを許可するディレクトリ (マウントポイント) とパーティションを限定する。
  - 書き込み可能領域に対しては noexec, nodev, nosuid オプションを強制的に有効にする。
    - バックドアを設置されたとしても実行させない。
  - chrootで移動可能なディレクトリを限定し、書き込み可能領域を / ディレクトリとして使用させない。
  - pivot_rootが不要になった時点で使用を禁止し、書き込み可能領域を / ディレクトリとして使用させない。



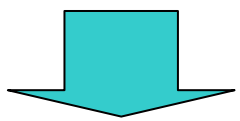
マウント制限の例 (dmesg出力)

```
% tail -10 /var/log/dmesg
sys_pivot_root() disabled.
Registered chroot dir: /jail/tomcat:/jail/apache:/var/empty/sshd
sys_chroot() restriction enabled.
Allow mount tmpfs on /data with options nosuid nodev noexec
Allow mount tmpfs on /jail/tomcat/data with options nosuid nodev noexec
Allow mount tmpfs on /jail/apache/data with options nosuid nodev noexec
Allow mount proc on /proc with options defaults
Allow mount proc on /jail/tomcat/proc with options defaults
Allow mount proc on /jail/apache/proc with options defaults
sys_mount() restriction enabled.
```

- 制限は起動スクリプトの中で自動的に有効となる。
- pivot_rootの使用を禁止。
- chrootは/jail/tomcat, /jail/apache, /var/empty/sshdに限定して許可。
- mountは /data, /jail/tomcat/data, /jail/apache/dataについて nosuid, nodev, noexec 指定を自動的に有効にした上で許可。
- HD等をマウントする場合はtmpfsの部分がdev=803 (/dev/sda3) のようになる。

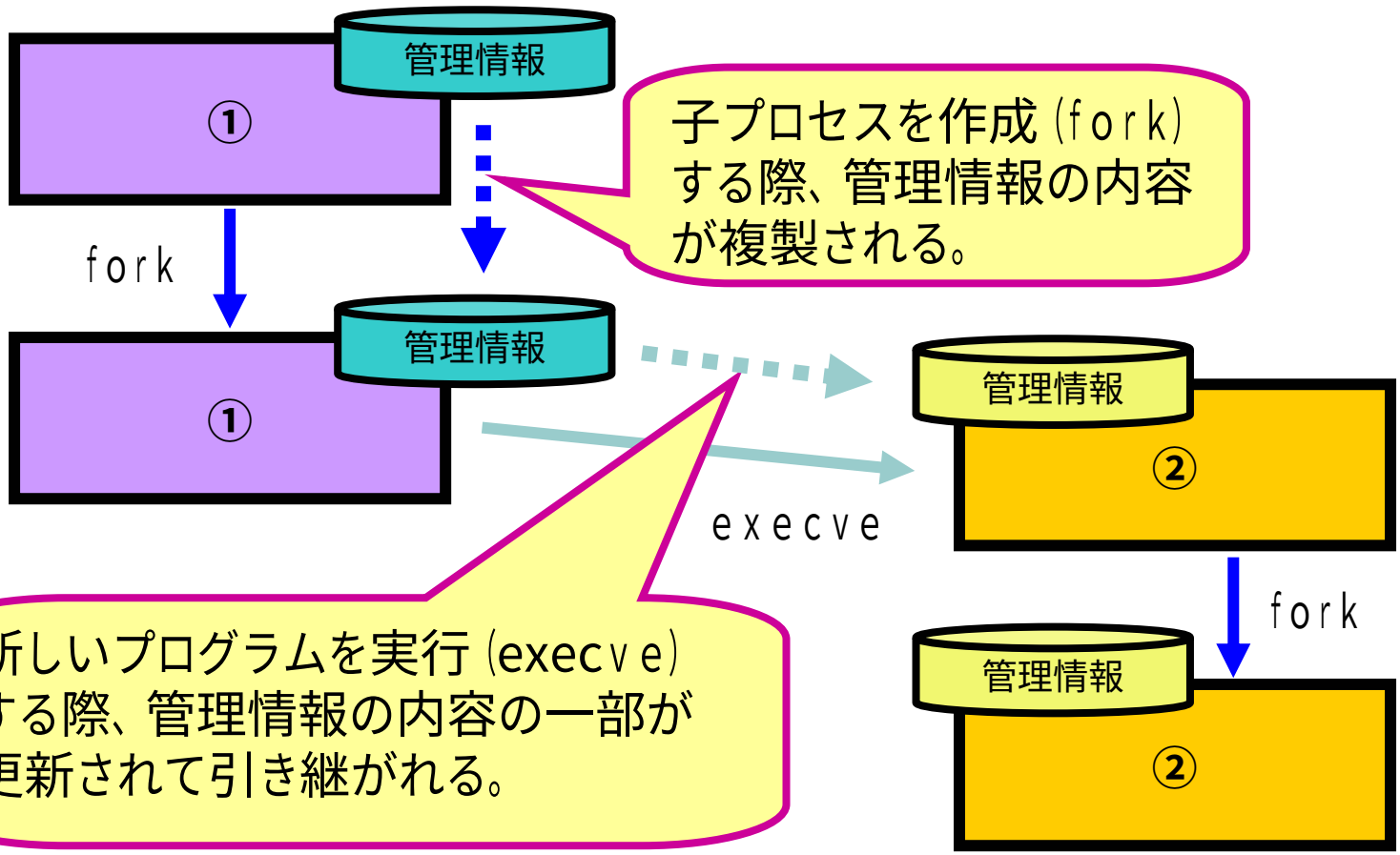
サービスの隔離による保護

- 公開サービスはchroot環境下で動作させる
  - chroot環境下には必要最小限のファイルしか置かない。
    - 「必要最小限」のファイルリストを作るためのカーネルパッチを作成
      - タスク構造体にメンバーを追加し、chrootされたプロセスとその子孫を他と区別
      - chrootされたプロセスについてアクセス要求を記録
      - アクセス要求のあったファイルのみを自動的に配置

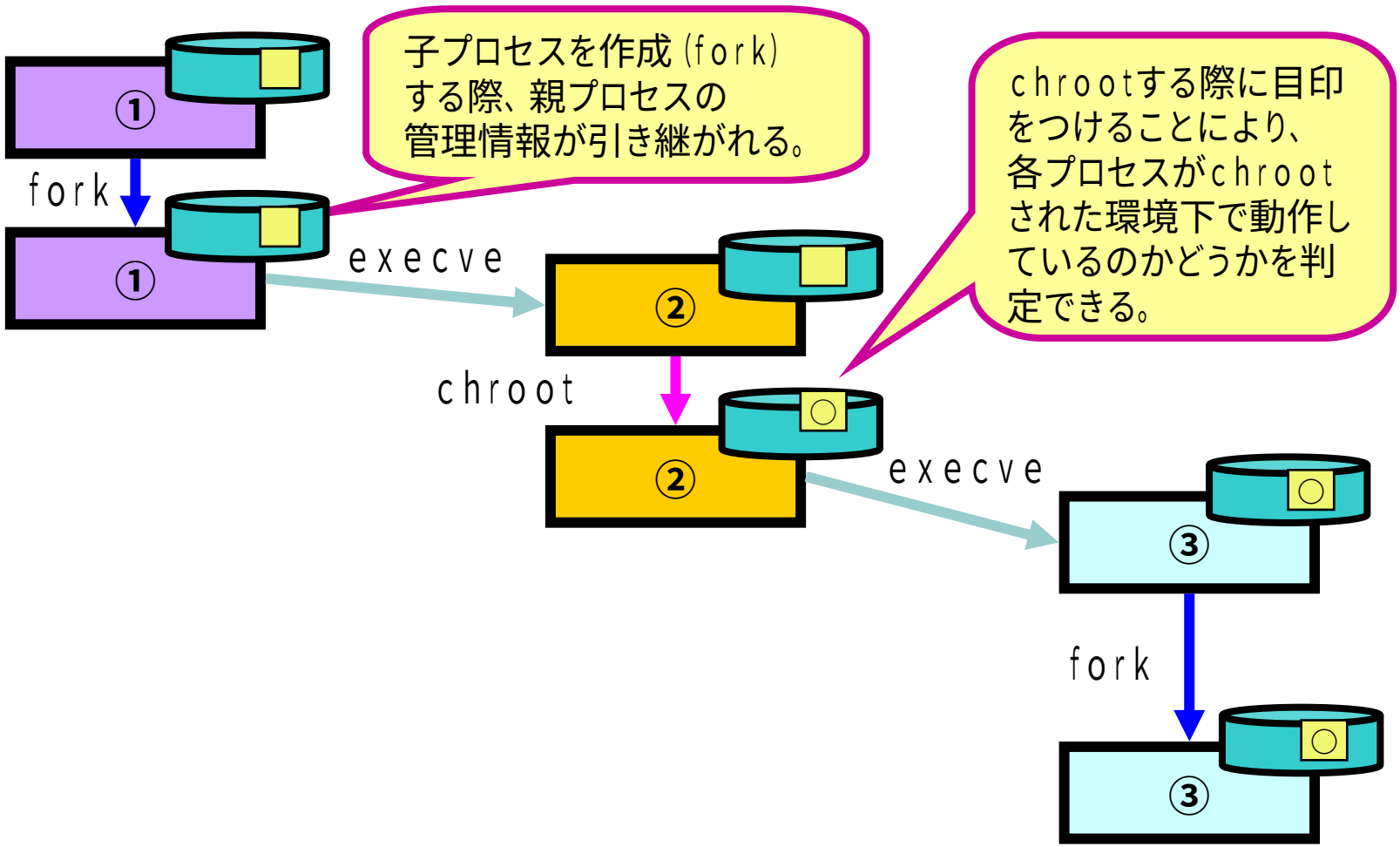


隔離環境の構築を全自動化した

プロセスの遷移

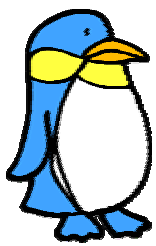


プロセスの追跡



タスクベースのアクセス制御

- コマンド自体が改ざんできなくてもバッファオーバーフロー等による乗っ取りは可能
  - カーネルにタスクベースのアクセス制御を実装
    - タスク構造体に「execve の使用禁止」フラグを追加
    - プロセスが「この pid を持つプロセスではこれ以上 execve が必要ない」と宣言すると、以降の execve をカーネルで禁止。プロセスが乗っ取られたとしても異なるイメージに切り替えられず、シェルやターミナル等を開始できない。
  - アプリケーションが自発的に権限を放棄することでセキュリティを向上させることができる。



タスクベースのアクセス制御の例

- 実行結果

```
[root@romlinux root]# bash
[root@romlinux root]# echo $$
966
[root@romlinux root]# date
金 10月 24 14:06:06 JST 2003
[root@romlinux root]# ./limitexec.exe $$
[root@romlinux root]# date
bash: /bin/date: 許可されていない操作です
[root@romlinux root]# ls
bash: /bin/ls: 許可されていない操作です
[root@romlinux root]# exit
exit
[root@romlinux root]#
```

シェル自身に対して、これ以降のexecveを禁止する。

- dmesg 出力

```
do_execve() for pid=966 disabled.
do_execve(pid=989:file=/bin/date): Permission denied.
do_execve(pid=990:file=/bin/ls): Permission denied.
```

子プロセスは生成されるが、execve が許可されていないので実行できない。

既存技術によるセキュリティ向上策

- 公開サービスをシステム管理者権限で動かしたくない。
  - カーネルのルーティング機能 (iptables) を使い、一般ユーザ権限でサーバを動かし、特権ポートでサービスを提供

```
[root@romlinux log]# for i in `pidof httpd`; do echo -n "Pid: " $i "  "; cat
   /proc/$i/status | grep "Uid:"; done
Pid:  853   Uid:   501      501      501      501
Pid:  852   Uid:   501      501      501      501
Pid:  851   Uid:   501      501      501      501
Pid:  850   Uid:   501      501      501      501
Pid:  849   Uid:   501      501      501      501
Pid:  848   Uid:   501      501      501      501
Pid:  847   Uid:   501      501      501      501
Pid:  846   Uid:   501      501      501      501
Pid:  824   Uid:   501      501      501      501
[root@romlinux log]#
```

既存技術によるセキュリティ向上策

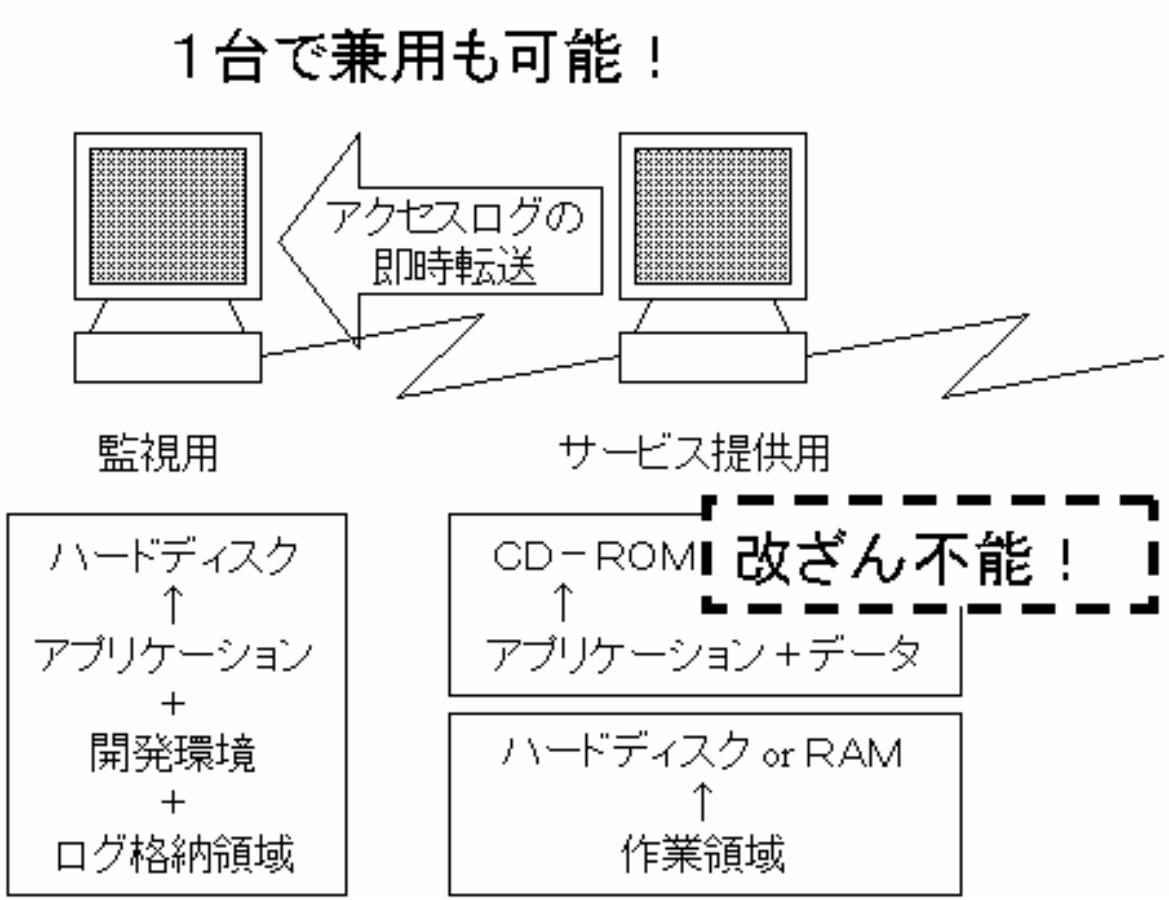
- システム管理者権限を奪われるとログの改ざんが可能となる
  - 「ログファイル」を書き換え不可能なメディアの中に FIFO として用意
    - 追記用途ではアプリケーションからは完全に等価（通常のファイルに見える）
    - chroot 環境の外側から FIFO を読み出してファイルに保存するので、chroot 環境下の公開サービスを乗っ取られたとしてもログの参照や改ざんは不可能
    - FIFO から読み出したログをネットワーク経由で他のサーバに転送することも可能

プロトタイプシステムの紹介

- 特長
  - 物理的に改ざんを防ぐ
  - バックドアを設置されても実行できないので安全
    - ただし、Java のクラスファイルやスクリプトファイル等は noexec による保護の対象外
  - CD-R でブート可能
    - Apache, Tomcat を含め「最小構成」で容量約 400MB (大容量が必要な場合は DVD-R か、ハードディスクの書き換え保護装置を利用可能)
    - 電源オンでサーバ起動、Ctrl+Alt+Del で再起動、HD を使っていなければいつでも電源をオフできる
    - バージョン管理が容易 (わかりやすい)
    - root fsのfsckが不要 :-)

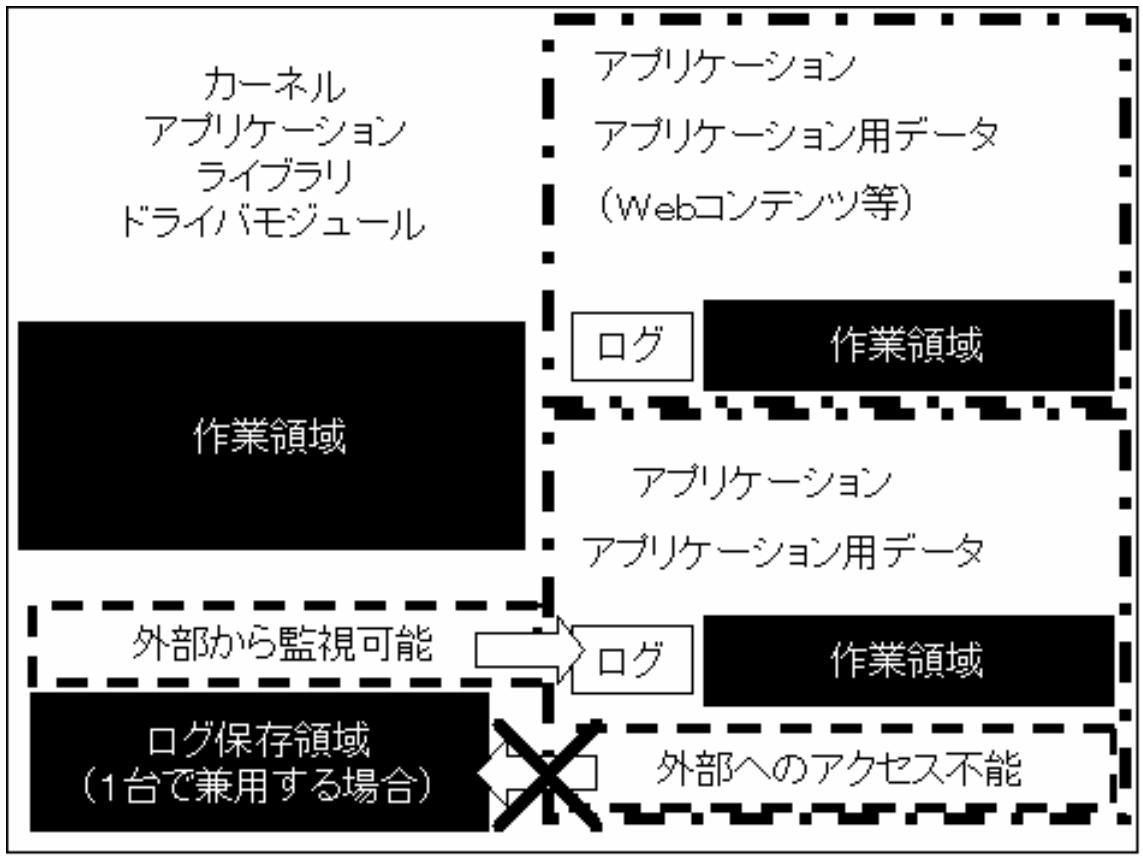
プロトタイプシステムの紹介

- 運用イメージ



プロトタイプシステムの紹介

- ファイル構成

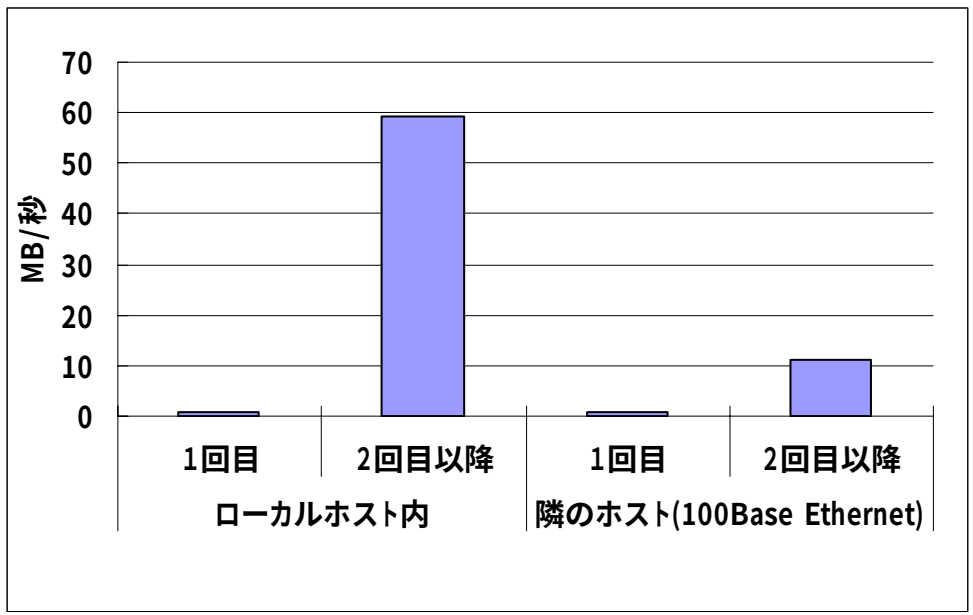


プロトタイプシステムの紹介

- 自動化できる部分はほとんど自動化済み
  - OSのインストールと再起動を除きほとんどの部分は HTMLで書かれた手順書をコピー＆ペーストするだけ。
  - Tomcat と Apache に関してはインストールから chroot 環境の構築まで全自動。
  - 運用環境の構築に要する時間は、インストール完了から CD-R への記録を開始するまで約３０分。システム管理に慣れた人なら２０分でも可能。
  - 今後社内サーバに適用し、評価する。

プロトタイプシステムの紹介

- 装置仕様
  - DELL PowerEdge 1550 （Pentium III 1GHz、1280MB、最大24倍速 CD-ROM ドライブ）
- Webコンテンツの転送速度



KNOPPIXとの比較

- 共通点
  - CD-ROMで起動すること
- 相違点
  - KNOPPIX
    - デモ、レスキューがメイン
    - cloop による豊富なアプリケーションと優れた自動認識
    - Debian/GNUベースの独自カーネル
  - 本システム
    - 改ざん防止が目的
    - モジュール、アプリケーションはあくまで最小限
    - Red Hat ベースで小規模、ポータブルなパッチにより実現
    - 「自分で作れる」改ざん防止サーバ

おわりに

- 「セキュアなＯＳ」は存在しない
  - いかにして考えられる脅威の範囲を限定し、それに対する被害を軽減するかが重要
- ＯＳセキュリティ強化のよりどころ
  - SELinux FAQより <http://www.nsa.gov/selinux/faq.html>
    - 「無修正の Linux のセキュリティは、カーネルの正しさ、全ての特権を与えられたアプリケーションおよびそれらの設定の正しさに依存する。」
    - 「SELinux のセキュリティは基本的には、カーネルの正しさとポリシーの定義の正しさに依存する。」
- 本システムのよりどころ
  - 本システムは、「物理的な保護」を基盤として、考えられる脅威に対する対策を積み上げたものであり、カーネルの正しさは無修正の Linux と同等で、ポリシー定義は不要。

参考文献

- 本研究を行うにあたり山森丈範氏の下記著作から多くの示唆を得ました。山森氏に感謝致します。
  - 「CD-ROMだけで動作するオリジナル Linux を作ろう」
    http://www15.big.or.jp/~yamamori/sun/cdlinux/
    Software Design 2002年12月号
- initrd, devfs の詳細について
  - devfs ドキュメンテーション
    http://www.atnf.csiro.au/~rgooch/linux/docs/devfs.html
  - /usr/src/linux/Documentation/initrd.txt
  - “Booting Linux: The History and the Future”
    - Werner Almesberger
- 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.nsa.org/nsf2003/
    原田季栄、保理江高志、田中一男