



セキュリティ強化Linux「TOMOYO Linux」の 商用システム導入事例紹介

2006年 5月15日

株式会社NTTデータ
基盤システム事業本部
オープンソース開発センター



はじめに

弊社が開発したセキュリティ強化Linux「**TOMOYO Linux**」の商用システムへの導入事例をもとに、**TOMOYO Linux** の機能・特徴と、セキュリティ強化Linux 導入のポイント・課題を紹介します。

※ **TOMOYO Linux** についてより深く知りたい方へ:

下記に全ての情報がありますのでご参照願います。

<http://sourceforge.jp/projects/tomoyo/>

<http://d.hatena.ne.jp/keyword/TOMOYO%20Linux>



本日のアジェンダ

- **第一章：**
セキュリティ強化OSを導入した背景とTOMOYO Linux採用まで
- **TOMOYO Linuxのデモ**
- **第二章：**
TOMOYO Linuxを使用したシステム構築
- **第三章：**
TOMOYO Linuxの運用
- **第四章：**
TOMOYO Linuxを採用したシステム構築のまとめと課題



第一章：

セキュリティ強化OSを導入した背景と TOMOYO Linux 採用まで



1-1. セキュリティ強化OS を導入した背景

■ お客様からの要求

- 個人情報扱うシステムであるため、絶対に不正アクセスは許されない
- システムが安全であるという保障、証明が求められる

■ 現実

- 通常のセキュリティ対策（ファイアウォール, 侵入検知システム, ウィルス対策など）の効果は限定的であり、それだけでは完全ではない
- 正規のプロトコルを用いた攻撃
 - OS、ミドルウェア、業務アプリケーションのバグに対する攻撃（**バッファオーバーフロー**など）

これらはプログラムの欠陥なので、根絶することは不可能



1-2. セキュリティ強化OS を導入した背景(つづき)

■ 一般的なセキュリティ対策

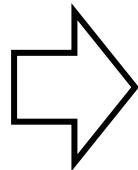
ファイアウォールやIDSで防げる脅威はネットワークに関連するものに限られる

サーバ内で行われる攻撃には対処できない

■ 迅速にパッチを適用して、OSやミドルウェアを最新に保つ

- パッチがリリースされるまでのタイムラグ
- パッチを適用するまでのタイムラグ
- 未知または未公表の脆弱性

0-dayアタックには対処できない



そこで セキュリティ強化Linux を採用



1-3. Linuxが標準で備えているアクセス制御機構

■ Linuxが標準で備えているアクセス制御機構

■ 任意アクセス制御(DAC)

- ファイルやディレクトリに、Owner, Group, Other に対するアクセス制限を設定
- システム管理者(root)は、上記の設定如何にかかわらず、全てのファイルやディレクトリのアクセス制限を変更や無視できる

→ システム管理者 (root) は神様

つまり

“バッファオーバーフロー” や “0-dayアタック” によって

不正にシステム管理者権限を奪取されると、オシマイ



1-4. セキュリティ強化Linuxとは

■ 代表的なものとしては「SE Linux」

- 我々が採用したのは「TOMOYO Linux」
- 基本となるのは、「強制アクセス制御(MAC)」機能
 - OSが管理する資源や機能に対するアクセス要求の諾否判定を、システム管理者でも回避できないように強制的に実施
 - 諾否判定はすべて「ポリシー」に従う

→ システム管理者であっても、行動に制限をうける

つまり

“バッファオーバーフロー” や “0-dayアタック” によって

不正にシステム管理者権限を奪取されても、適切なポリシーが定義されている限り、大丈夫（影響が把握可能）



1-5. 利用者側が求めるセキュリティ強化OS

■ 求めていたのは、目的にあったセキュリティ強化OS

- 本事例では、侵入防止が主目的
- 実システム適用にあたっては、OSがセキュアであるだけでは不十分。実際に「使いこなせる」ことを重視した
 - システムを構築・運用するメンバーはセキュリティ専門家ではない
 - 運用期間中(5年間)に、構成変更が発生する可能性が高い
 - ディストリビューション標準以外のミドルウェアを使用する

■ 利用者の立場から見た SE Linux

- SE Linux は、セキュリティ・ラベルをベースにしているため、ラベルの設計とファイルへの対応付けが必要
- ポリシー言語の粒度が非常に細く、直感的には理解できない
- 一般にはデフォルトポリシーをそのまま使用する形態が多い

→ オリジナルのポリシー定義が難しい
ポリシー自体も複雑で理解が難しい



1-6. なぜ TOMOYO Linux を採用したのか

■ TOMOYO Linux とは

- 最大の特長はポリシー自動学習機能（後ほどデモします）
 - プロセスの起動履歴毎にドメインを割り当てアクセスを制御
- ドメインをパス名で指定できるため、理解しやすい
- アクセス制御内容も標準的なスキルで十分理解可能（読める、わかる）

■ 利用者の立場から見た TOMOYO Linux

- ポリシー自動学習機能が便利
- ドメインをパス名で指定できるため、理解しやすい
- ポリシー言語の粒度は、必要なセキュリティ強度を保ちつつ、SE Linux ほど細かい（その反面、SE Linux ほど細かな制御はできない）
- 構築・運用を支援する各種機能（信頼済みドメインなど）

→ オリジナルのポリシー定義が簡単

ポリシー自体もシンプルで誰でも直感的に理解できる



1-7. TOMOYO Linux と SE Linux の比較(ポリシー)

■ TOMOYO Linuxのポリシー例

```
<kernel> /usr/sbin/sshd /bin/bash
          1          /usr/bin/passwd
<kernel> /usr/sbin/sshd /bin/bash /usr/bin/passwd
          6          /etc/shadow
```

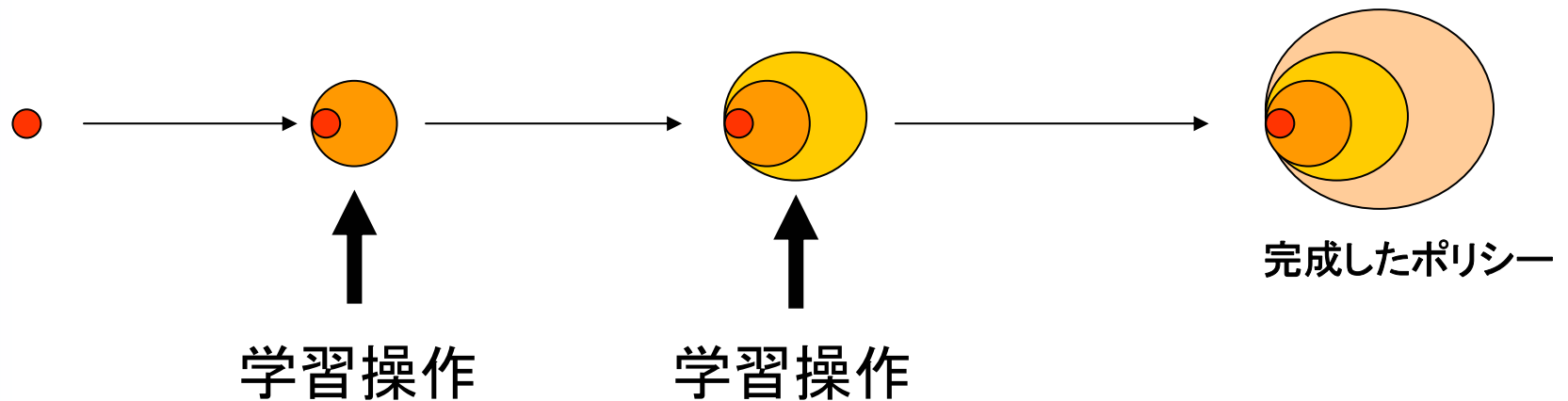
■ 上記を「SE Linux風」に記述したとすると... (ファイルコンテキスト定義は除く)

```
type "<kernel> /usr/sbin/sshd /bin/bash", domain;
type "<kernel> /usr/sbin/sshd /bin/bash /usr/bin/passwd", domain;
allow "<kernel> /usr/sbin/sshd /bin/bash"
    "/usr/bin/passwd": file execute;
allow "<kernel> /usr/sbin/sshd /bin/bash /usr/bin/passwd"
    "/usr/bin/passwd": file entrypoint;
allow "<kernel> /usr/sbin/sshd /bin/bash"
    "<kernel> /usr/sbin/sshd /bin/bash /usr/bin/passwd":
    process transition;
allow "<kernel> /usr/sbin/sshd /bin/bash /usr/bin/passwd"
    "/etc/shadow": file { read write };
```



1-8. TOMOYO Linux と SE Linux の違い(ポリシー作成)

■ TOMOYO Linux 基本的にポリシーの全てを学習させる



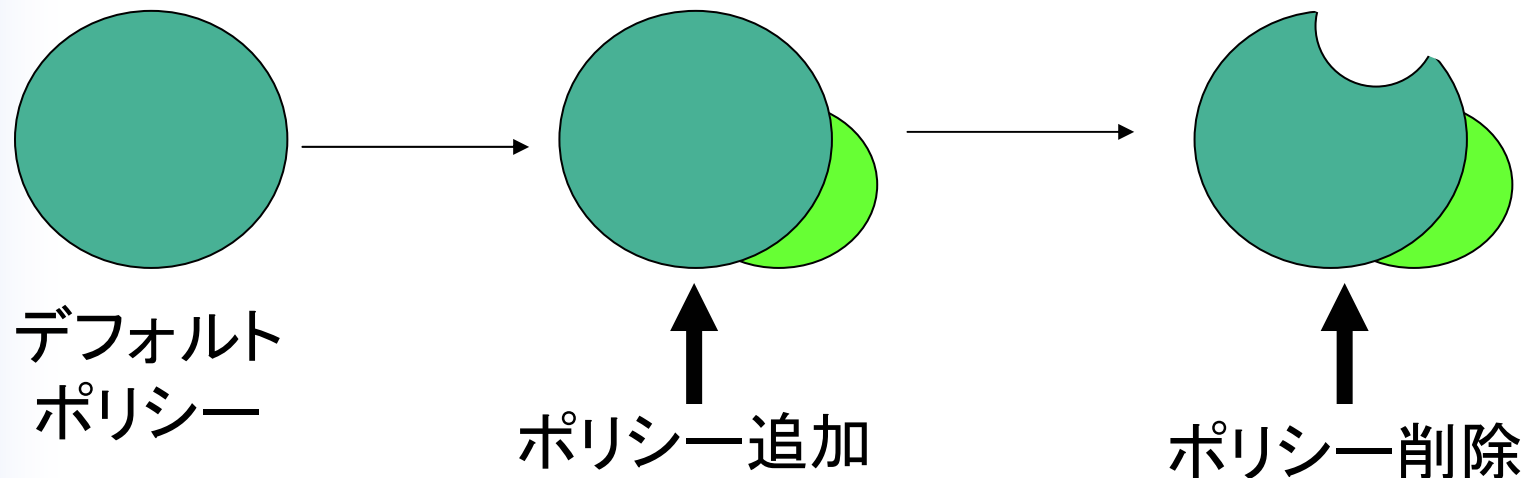
長所: 学習した事以外(=想定外)の事象は発生しない
→ “学習操作=システムのなふるまい”なので
専門家以外でもポリシー定義が可能

短所: 実運用に入り、学習外の事態が発生すると即障害になる



1-9. TOMOYO Linux と SE Linux の違い (ポリシー作成)

■ SE Linux 基本的に用意済みポリシーをカスタマイズする



長所: ベースとなるポリシーが存在する

短所: ポリシーに不必要な部分が残存する可能性がある

追加したポリシーとデフォルトポリシーの整合性保障が困難

→ 専門家でないとポリシー定義が難しい



1-10. TOMOYO Linux と SE Linux の向き不向き

■ 利用者の立場から見た向き不向き

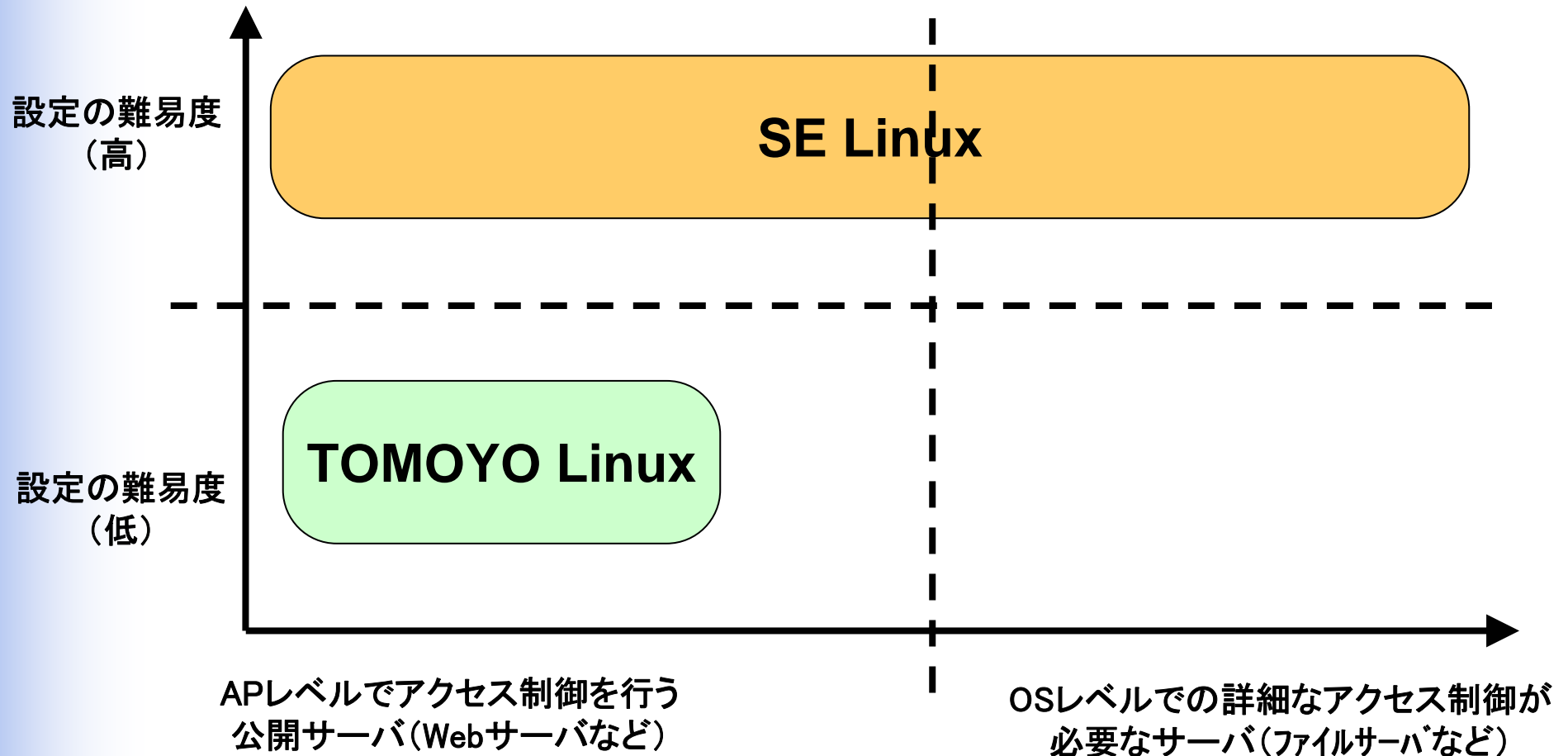
- 運用フェーズで、**設定の変更が必要**である場合
- ポリシー定義を**専門家以外**が行う場合
- **ロール単位での制御が不必要**な場合
→ TOMOYO Linux

- 運用フェーズで、**設定の変更が不必要**である場合
- ポリシー定義を**専門家**が行う場合
- **ロール単位での制御が必要**な場合
→ SE Linux



1-11. TOMOYO Linux と SE Linux の向き不向き(イメージ)

- 相互に補完的であり、一方が他方を凌駕する関係ではない





1-12. TOMOYO Linux のデモ

■ TOMOYO Linux の簡単なデモ

- ポリシーの自動学習
- ポリシーの可読性の良さ
- ポリシーに基づく、強制アクセス制御

TOMOYO Linux のメリットについて、体験してみましよう。



第二章：

TOMOYO Linuxを使用したシステム構築



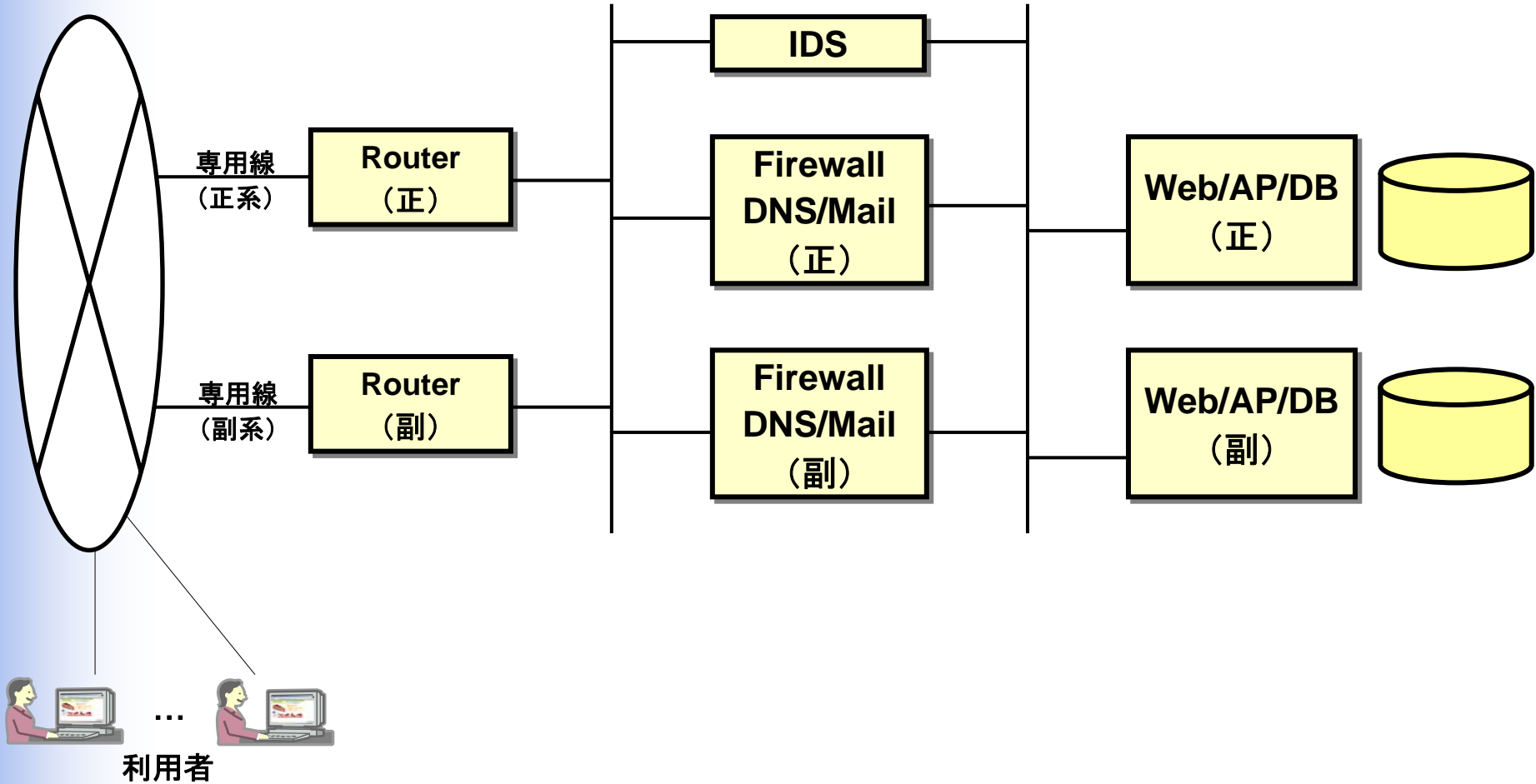
2-1. TOMOYO Linux ベースのシステム構築

- セキュリティ強化OS のメリットは理解され始めているが、実際に導入したという事例は少ない
- 事例を通じて、その原因と我々が実施した対策についてご紹介します



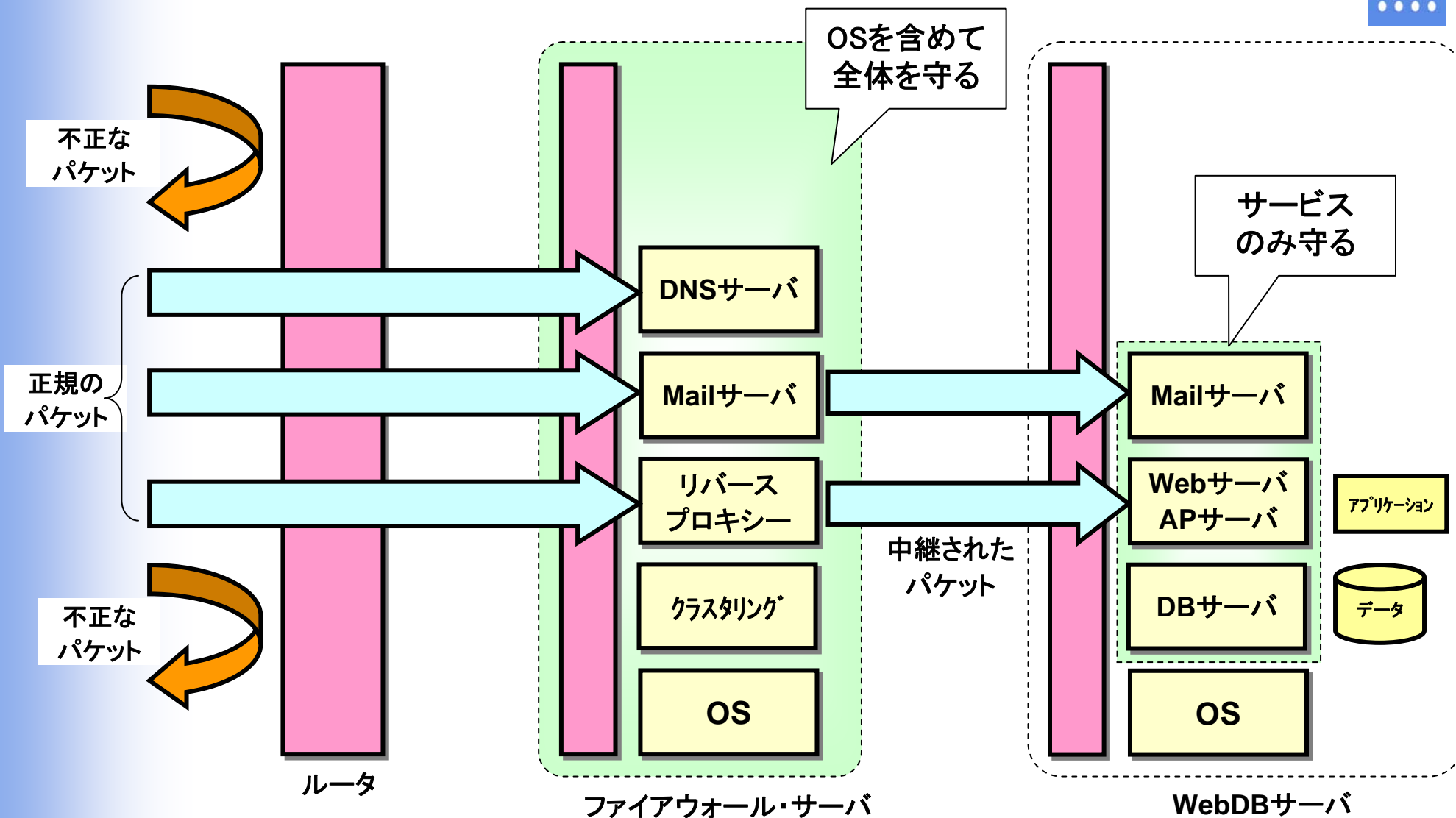
2-2. システム構成(ハードウェア)

インターネット





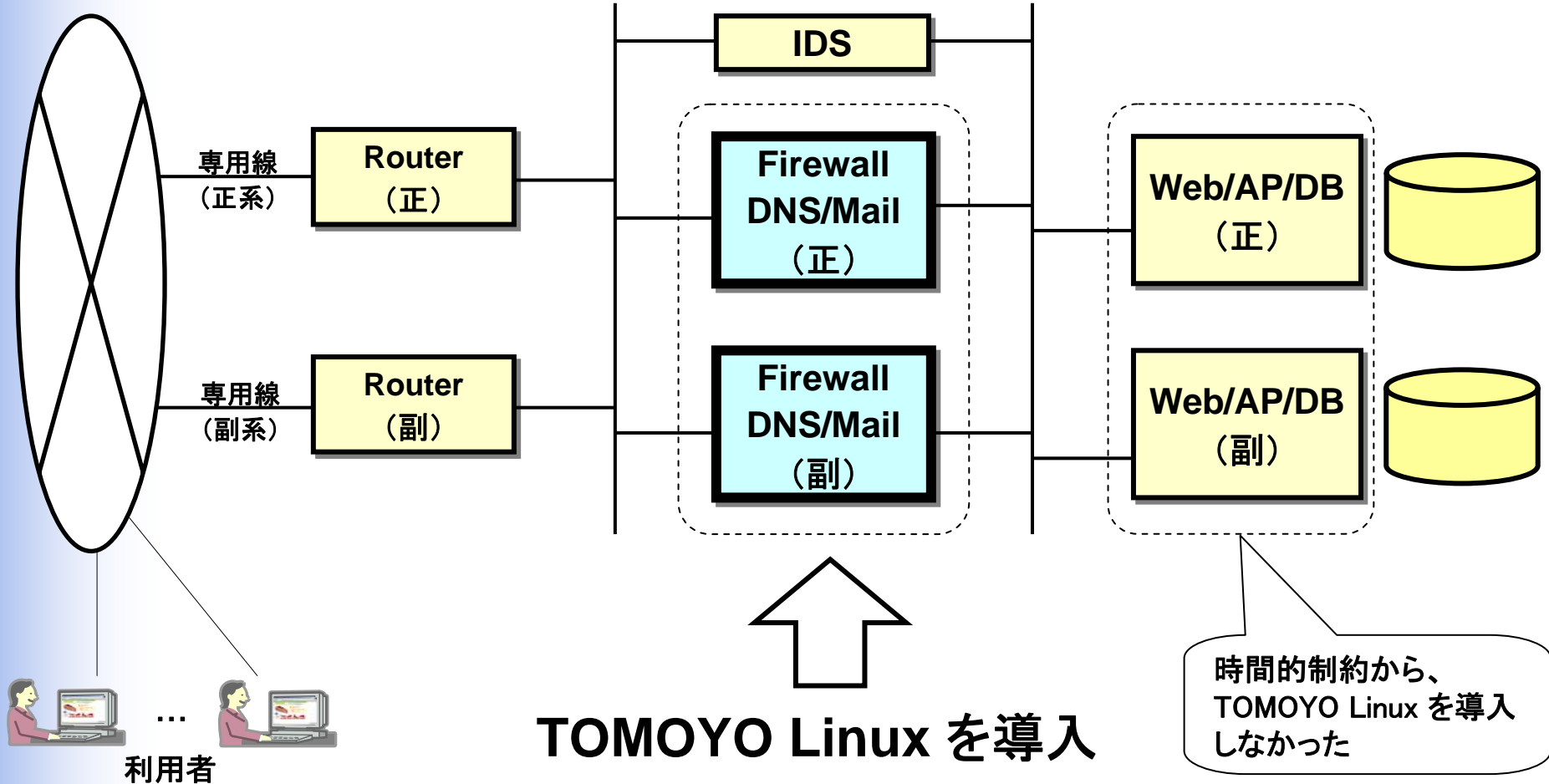
2-3. 守るべき場所





2-4. システム構成(ハードウェア)

インターネット





2-5. システム構成(ソフトウェア)

■ TOMOYO Linux でアクセス制御する対象 (抜粋)

項目	導入ソフトウェア
OS	FedoraCore3 (kernel-2.6.11)
リバースプロキシ	Pound
DNSサーバ	BIND
メールサーバ	Postfix
クラスタリング	Heartbeat
運用・監視系エージェント	cron net-snmp AFbackup apcupsd sshd



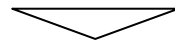
2-6. セキュリティ強化OSを使用すると何が違うのか？

■ 最大かつ唯一の違いは、「ポリシー」の有無

- TOMOYO Linux であれ、SE Linux であれ、ポリシーを定義する必要がある
- 全てはポリシーの質で決まる

<セキュリティ強化OSの導入が進まない原因1>

- ポリシーをゼロから定義するのは難しい
⇒ いかにして質の良いポリシーを簡単に定義するか？



**自動学習によるポリシーの定義結果から
始められるのがTOMOYO Linuxの利点**

- 次のスライドから、ポイントを紹介していきます
- TOMOYO Linux についての事例ですが、その他のセキュリティ強化OSに対しても、同様な考え方もできると思います。



2-7. ポリシー定義のポイント1



1. 事前準備

■ あたりまえのセキュリティ対策

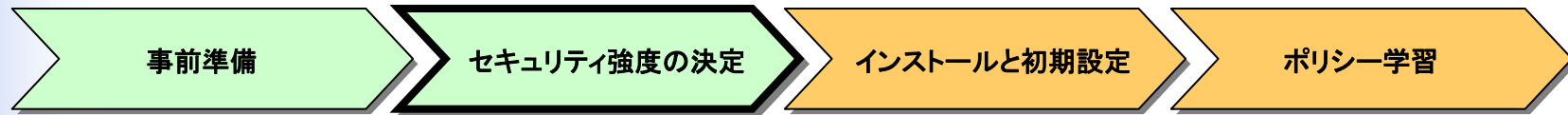
- 不要なポートの遮断
- 不要なサービスの停止
- 不要なアカウントの削除
- 不要なパッケージの削除
- 最新セキュリティアップデートの適用

■ ポリシー学習環境

- “学習”なので、実機環境が必要
- まず、開発環境や仮想マシン（VMwareなど）で学習ノウハウを習得
- ハードウェアが異なるとデバイス名が異なる（＝ポリシーも異なる）ため、最終的には本番機で学習する



2-8. ポリシー定義のポイント2

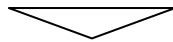


2. セキュリティ強度の決定（網の目の細かさを決める）

- ポリシーは厳しければ良いというわけではない
 - セキュリティ強度と利便性（運用性）のトレードオフ
 - 厳しければ厳しいほど、ポリシー定義に必要な工数が増える
- トラブル対応時の操作など、事前に予測が難しいものへの対処

<セキュリティ強化OSの導入が進まない原因2>

- 導入対象サーバ、守るべきサービスが決められない



1. インターネットから直接的に攻撃をうけるサーバ
→ 包括的なポリシー（OS, サービス全てのポリシーを定義する）
2. それ以外
→ 部分的なポリシー（守るべきデーモンのみポリシーを定義する）



2-9. TOMOYO Linux のインストールと初期設定



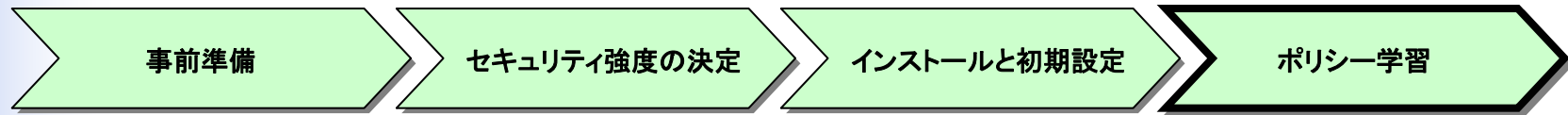
3. TOMOYO Linux のインストールと初期設定を行います

下記のドキュメントに、TOMOYO Linux のインストールと初期設定について丁寧に解説してあります。

<http://tomoyo.sourceforge.jp/ja/doc/install.html>



2-10. ポリシー定義のポイント3(学習モード)

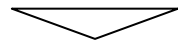


4. 学習モードでポリシーの学習を行う

- ポリシー学習方針に基づき、ポリシー学習操作をおこなう
- 操作の度にポリシーファイルの内容をチェックし修正を繰り返す

<セキュリティ強化OSの導入が進まない原因3>

- ポリシー学習の方針が決められない

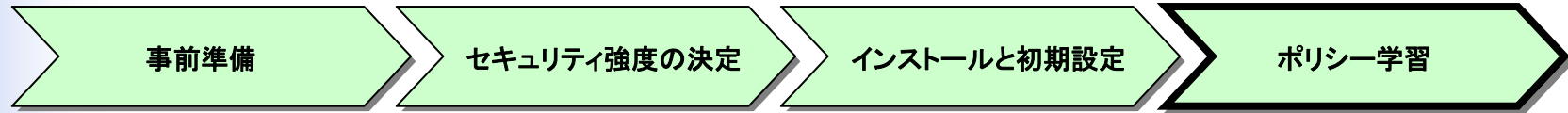


■ ポリシー学習方針の例 (TOMOYO Linux限定)

- サーバ起動時、シャットダウン時の挙動
 - コマンド操作に加えて、UPS管理デーモンからのシャットダウンを学習
- サービス起動スクリプトの操作
 - /etc/init.d 配下で使用するスクリプトを実行(start/stop/reloadなど)
- cron から起動されるジョブ
 - サーバ時刻を変更しつつ、全て(hourly/daily/weekly/monthly)学習



2-11. ポリシー定義のポイント4(学習モード:つづき)



■ ポリシー学習方針の例 (TOMOYO Linux限定)

- デーモン類への操作 (Postfix、BIND、Pound、net-snmp、Heartbeat など)
 - シンプルなもの (ゾーン転送しないBINDなど)
 - 起動時に設定ファイルを読み込み、ログを出力するだけの動作が多い
 - エラーログ
 - エラーログが通常ログと別ファイルに分かれている場合、ポリシー学習のために人為的にエラーを発生させる
 - テンポラリファイル
 - 毎回ファイル名が変更されるため、ワイルドカードで指定する。
 - 単機能のプログラムが協調して動作する構成のミドルウェア (Postfixなど)
 - マイナーな機能を受け持つプログラムについてポリシー学習が漏れがちなので注意する
 - エージェント類
 - マネージャーからリクエスト送り、ポリシー学習する



TOMOYO Linux ポリシー例 (afbackup)

```
<kernel> /usr/sbin/xinetd /usr/local/backup/server/bin/afserver
```

```
2 /dev/null
4 /dev/random
4 /etc/localtime
4 /etc/resolv.conf
4 /lib/libacl.so.1.1.0
4 /lib/libattr.so.1.1.0
1 /usr/local/backup/client/bin/full_backup
4 /usr/local/backup/server/etc/backup.conf
4 /usr/local/backup/server/lib/keyfile
6 /usr/local/backup/server/var/pref_client
2 /usr/local/backup/server/var/server.backup.log
2 /usr/local/backup/server/var/status
6 /usr/local/backup/server/var/tapepos
2 /usr/local/backup/server/var/tapepos.tmp
```

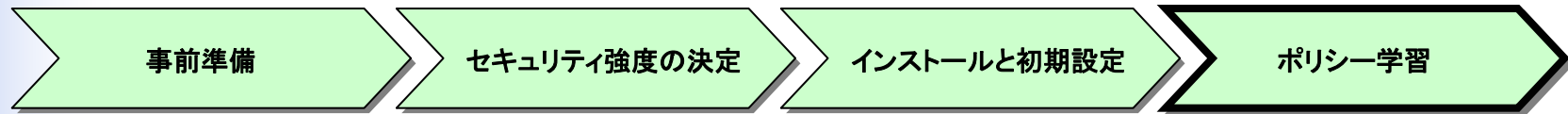
```
<kernel> /usr/sbin/xinetd /usr/local/backup/server/bin/afserver /usr/local/backup/client/bin/full_backup
```

```
1 /bin/bash
1 /bin/gunzip
1 /bin/gzip
4 /dev/random
6 /dev/tty
4 /etc/localtime
4 /etc/mtab
4 /etc/resolv.conf
4 /lib/libacl.so.1.1.0
4 /lib/libattr.so.1.1.0
6 /tmp/afbrep_¥*
2 /tmp/afbsp_¥*
1 /usr/local/backup/client/bin/afclient
4 /usr/local/backup/client/etc/backup_disk.conf
4 /usr/local/backup/client/etc/backup_tape.conf
2 /usr/local/backup/client/var/¥*
6 /usr/local/backup/client/var_disk/¥*
6 /usr/local/backup/client/var_tape/¥*
6 /var/lock/Lck.afbu_client.^usr^local
```

ポリシー全体: 約5,000行
544ドメイン



2-12. ポリシー定義のポイント5(学習モード:つづき)



■ 状態遷移の多いミドルウェア (Heartbeat)

- Active/Standby の HAクラスタ構成
 - クラスタリングの状態によって、動作が異なる
 - Active → Active, Active → Standby, Standby → Active, Standby → Standby
 - 自分自身がトリガ、相手方がトリガ、プロセス監視デーモンがトリガ
- 4 × 3 = 12通り

インターネットからのアクセスを待受けないデーモンに関しては、信頼済みドメインに入れることも考慮する(適切な網の目)

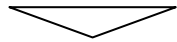
※信頼済みドメイン: 強制アクセス制御を適用しないドメイン



2-13. ポリシー定義のポイント5(学習モード:つづき)

<セキュリティ強化OSの導入が進まない原因4>

■ ポリシーの質に自信がもてない



■ はじめから完璧は求めないこと

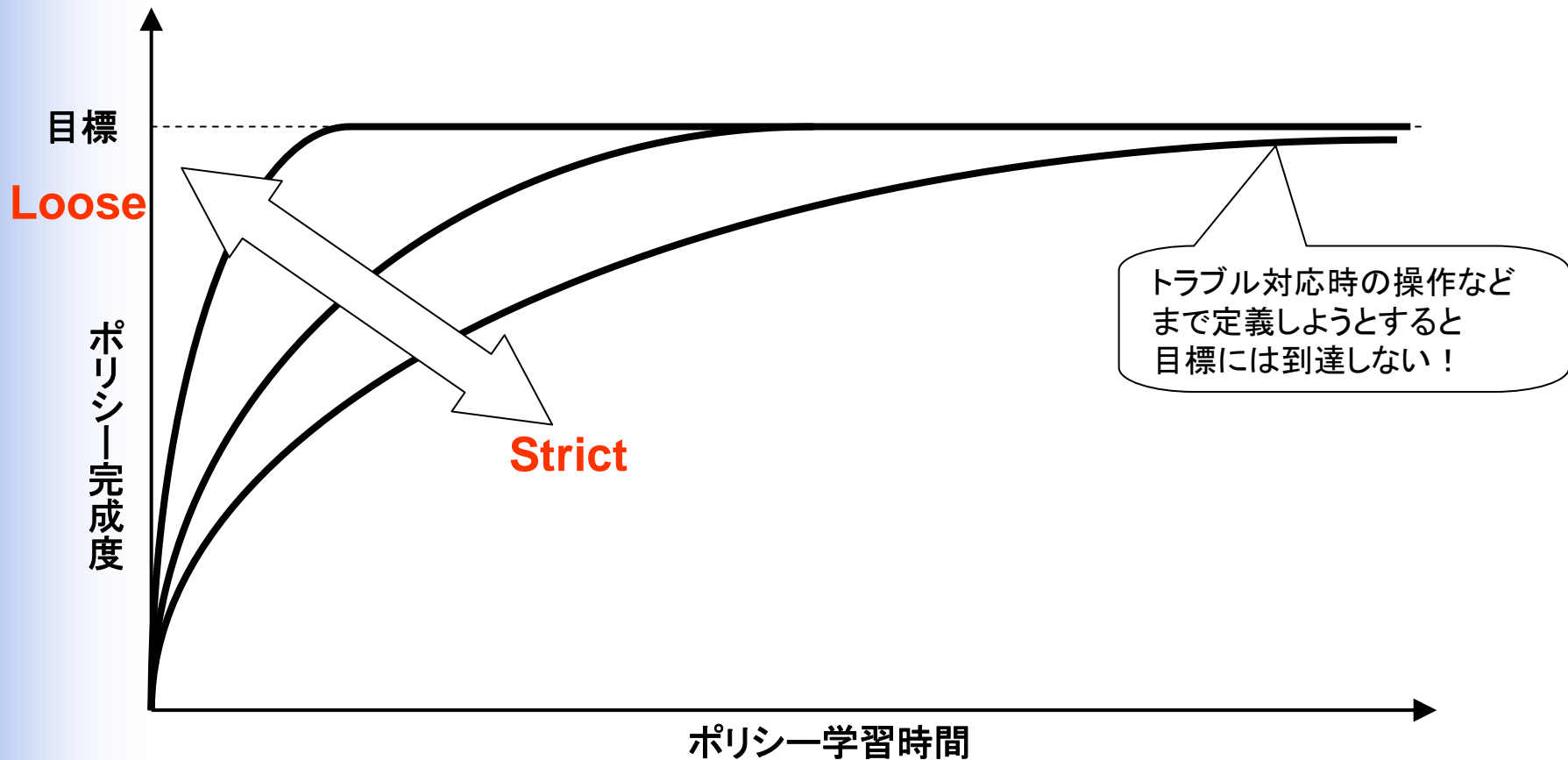
- 完璧を求めると、ミドルウェアのソースをチェックするしかない
 - ソースを「読める」と「理解できる」は別物
 - TOMOYO Linux の学習ログを見て、納得いかないものだけチェックする
- ときには SE Linux のポリシーも参考にする
- パターン指定や信頼済みドメイン機能を有効に活用する
- 商用ミドルウェアの場合 BlackBoxなので、網の目を緩くせざるをえない

■ システムの挙動(=ポリシー学習手順)を把握する

- ポリシー学習手順を残す
- ポリシーを読み、内容を理解する
 - ドメイン遷移の理解
 - 例外ポリシーの理解



網目を過剰に細かくすると、時間をかけても目標には到達しない

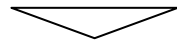




2-14. 構築期間

<セキュリティ強化OSの導入が進まない原因5>

■ 導入稼動が予測できない



■ システム基盤（プラットーホーム）部分

- 構成検討から構築、試験までトータルで 3ヶ月
- TOMOYO Linux 関連の作業は、そのうち 2ヶ月
→ 他作業と平行して作業したため、TOMOYO Linux による純増は3週間・人

■ 検討時（2005.3）では、まだ OSS公開前のバージョン

- その時点のバージョンでも必要としている機能は備わっていた
- TOMOYO Linux 最新版は さらに 実際の導入 & 運用を手助けするための機能やツールが充実している

現在の構成と同じという事が大前提であるが、今なら通常工期+2週間で可能



第三章：

TOMOYO Linux の運用



3-1. TOMOYO Linux の運用は何かちがうのか？(その1)

■ 運用期間

- 2005年7月 ~ 現在(約11ヶ月)
- 月次でセキュリティ報告を実施中

■ アタック監視

- セキュリティ強化OS未導入のサーバでは、アタックがあったかどうか確認する労力は非常に大きく、かつ完全とはいえない。
 - 複数のログファイルを常時チェックする必要がある
 - 本当に侵入されてしまった場合、痕跡が消されているかもしれない
- TOMOYO Linux が導入されていれば、違反ログを監視していれば良い
 - TOMOYO Linux での攻撃検出なし“0件”

→ TOMOYO Linux 導入によって、**アタック監視負荷は激減している**



3-2. TOMOYO Linux の運用は何かちがうのか？(その2)

■ セキュリティアップデート

- セキュリティホールに迅速に対応する必要がある
- パッチを適用するということは、デグレード確認試験を実施する必要がある
 - 運用期間中(約9ヶ月)にリリースされたセキュリティパッチは79個
 - 毎月膨大な稼動が発生する！
- TOMOYO Linux が導入されていれば
 - DoS攻撃以外のセキュリティホールは静観できる！

(例) 「Apache に巧妙なリクエストを送ることによって任意のコマンドを実行できる」という重大なセキュリティホールを利用して、シェル(/bin/bash)の実行を要求された場合

“<kernel> /usr/sbin/httpd /bin/bash” というポリシーが定義されていない限り、Apacheをのっもられても/bin/bashは起こされない！ =安全！

→ TOMOYO Linux 導入によって、**神経質にセキュリティアップデートを適用する必要はなくなり、運用の負荷は激減する**



3-3. TOMOYO Linux の運用は何がちがうのか？(その3)

■ 構成変更

- ポリシー定義の工夫によって、ある程度までの変更は許容できる
- ポリシーを修正する可能性が高いパターン
 - ハードウェア構成変更(NIC追加など)
 - ソフトウェア構成変更(ミドルウェアの追加導入、バージョンアップなど)
 - コンテンツ修正(Webサーバ、APサーバに導入した場合など)
- 結局はケースバイケースであり、稼動予測は難しい

→ TOMOYO Linux 導入によって、**構成変更やコンテンツ修正対応稼動は増大する**(SE Linux を導入していた場合よりは対応は楽)



3-4. TOMOYO Linux の運用は何かちがうのか？(その4)

■ まとめ

- TOMOYO Linux 導入で、運用負荷が軽減するもの
 - アタック監視
 - セキュリティアップデート
- TOMOYO Linux 導入で、運用負荷が増大するもの
 - 構成変更(ハードウェア、ソフトウェア)
 - コンテンツ修正(WebサーバやAPサーバ)

■ メリット > デメリット にするためには

- どのサーバに導入するか？(まずはファイアウォールサーバを推奨します)
- 適切な網の目(Strict からスタートし、運用性を向上させていく)
- 守るべきデーモンを限定する (Targeted なポリシーでの運用)
- TOMOYO Linux に適した領域に導入する



第四章：

TOMOYO Linux を採用したシステム構築 のまとめと課題



4-1. TOMOYO Linux を採用したシステム構築のまとめと課題

1. セキュリティ強化 Linux

- その必要性は今後ますます増大するだろう
- 求められるのは、「使いこなせて安全」
- メリット・デメリットを理解した上で、目的にあったセキュリティ強化Linux を選定（適材適所）することが重要

2. TOMOYO Linux

- TOMOYO Linux は、抜群に使いやすい
- 本事例の領域であれば実績もあり、展開が可能
- 構成によっては、運用コスト削減分で導入コストを回収することも可能
- SE Linux とは適応領域が異なる（相互に補完的な関係であり、一方が他方を凌駕する関係ではない）

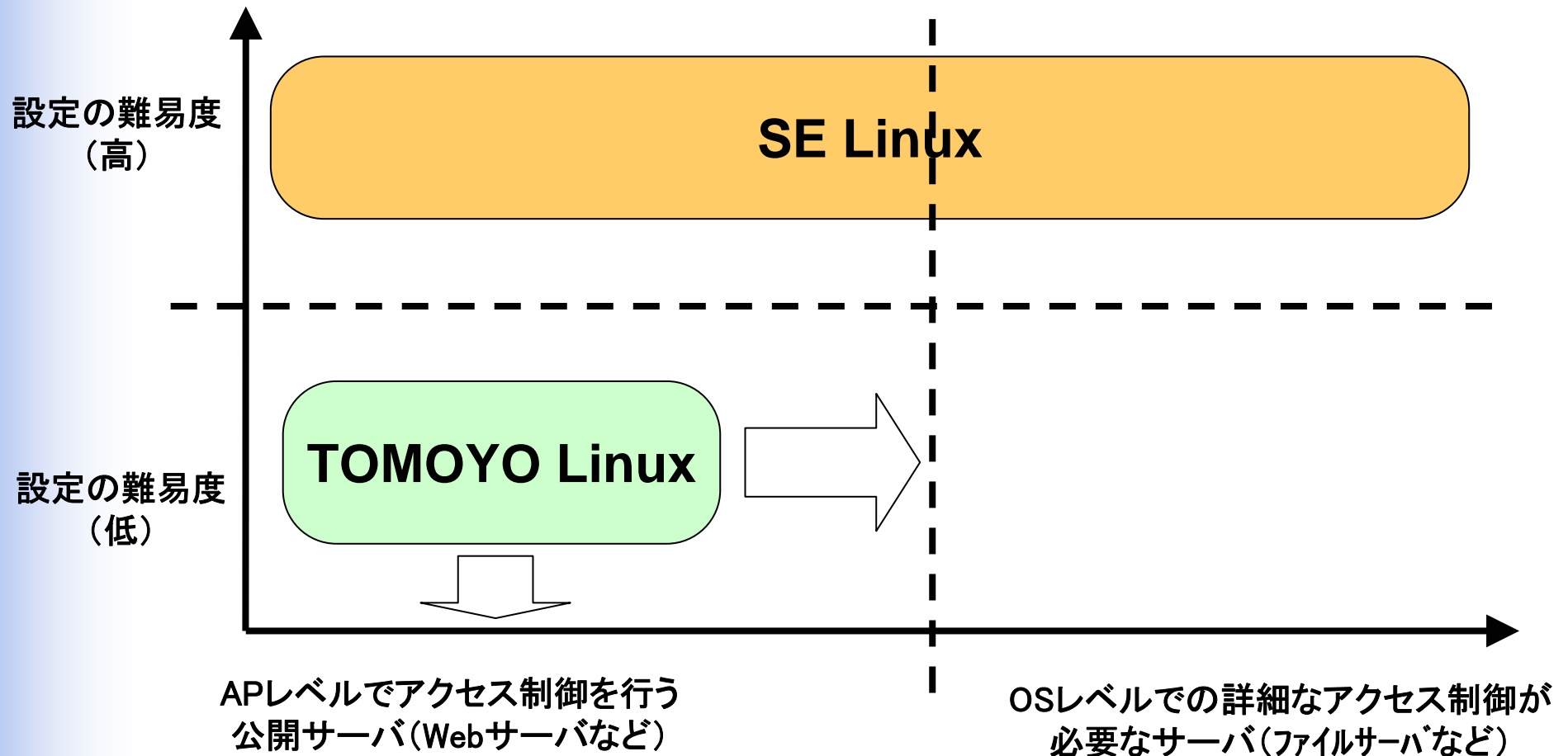
3. 課題

- 事例の積み重ねによる、対象領域の拡大
- さらなる導入コスト、運用コストの低減（アプライアンス化など）



4-2. TOMOYO Linux を利用した SI の発展イメージ

■ 得意とする領域内でカバーする範囲を広げていく





ご清聴ありがとうございました

本事例に関する問い合わせ先:

基盤システム事業本部 オープンソース開発センタ
OSSインテグレーションSU インテグレーション担当
岩田浩 <iwatahrs@nttdata.co.jp>

TOMOYO Linux に関する問い合わせ先:

基盤システム事業本部 オープンソース開発センタ
技術開発担当
原田季栄 <haradats@nttdata.co.jp>

TOMOYO Linux プロジェクト:

<http://sourceforge.jp/projects/tomoyo/>