平成15年11月25日　情報通信技術研究会説明資料

Linuxのセキュリティ強化について

株式会社ＮＴＴデータ
技術開発本部
原田季栄
haradats@nttdata.co.jp

本資料の位置づけ

- この後説明する Network Security Forum 2003 で発表した「プロセス実行履歴に基づくアクセスポリシー自動生成」を理解する上で必要な内容の解説（補足）です。
  - http://www.jnsa.org/nsf2003/
- 本資料単独で、「Linux のセキュリティ強化」について全体像を理解していただくことを目的として作成しました。
- 具体的には以下のトピックを含みます。
  - 「Linux のセキュリティを強化しなければならない理由」
  - 「具体的な強化方法と実装例」
  - 「アクセスポリシーとは何か」
  - 「アクセスポリシーの自動生成を試みた理由」
  - 「何故 SELinux は作られたのか、SELinux で十分なのか」

Linuxなら安全か？

- 知られているセキュリティ上の問題
  - root ユーザのプロセスは「全能」
    - アクセス制御を受けない
    - ファイルやディレクトリの所有者やアクセス制御内容を変更できる
    - どんなユーザのプロセスにも変身することができる
  - 貧弱かつ自由裁量のアクセス制御
    - 粒度
    - 所有者（かrootユーザ）による設定変更
- Linuxに対する攻撃はrootユーザ権限を取得することがゴールとなる。
  - root 権限のシェルを起動したら後は好き放題
- ファイアウォールや IDS の効果は？

ファイアウォールやIDSの役割と限界

- ファイアウォール
  - 「不要」な通信の遮断。
  - 使用（公開）しているサービスへのアクセスを閉ざすことはできない。
- IDS（侵入検知）
  - 登録されているパターン（シグナチャ）との照合による監視。
  - 未登録の攻撃は検知できない。
  - 検知しても直接被害を防ぐことはできない。
- （実は）効果は非常に限定的なもので、それだけで安心するのは間違い

どう攻撃されるのか？

- デモ
  - 脆弱 ftp サーバの乗っ取り
- 何が起こったか
  - ftp サーバに ftp クライアントとして接続
  - バッファオーバーフローを発生させる
  - root 権限の shell を起動
  - クラッカーはネットワーク経由でターゲットシステムの管理者となる

攻撃はどうすれば防げたのか？

- ネットワークに接続しない
  - 不可（セキュアだがサービスもできない）
- ファイアウォール、ＩＤＳ を導入する
  - 不可（導入していても被害は回避できていない）
- プログラムをアップデートし、ＯＳ にパッチを当てる
  - 有効（但し新規の脆弱性に対しては効力なし）
- バッファオーバーフローを防ぐ
  - 困難（入力を受け付けるプログラムの不具合根絶は不可能）
- プログラムをシステム管理者権限で動かさない
  - 困難
    - Ｌｉｎｕｘ は管理者権限で動作するプロセスを前提として設計されている。
      例：パスワードの更新、特権ポートの使用
    - カーネルの脆弱性は、一般ユーザ権限のプロセスの管理者権限への昇格を可能とすることがある
- つまり、標準の Ｌｉｎｕｘ では根本的な対処は存在しないということ

ではどうするか？

- サーバ自身が最後の砦として、サービスやデータを守ることが必要
  - http、ftp 等公開しているサービスからの攻撃に備えて
  - 内部からの不正アクセス、情報漏洩に備えて
    - 「情報漏洩の60%は内部犯行」
  - 万一被害を受けた時の解析、回復を可能とするため
    - root権限を奪われた場合、ログも信頼できない（信頼すべきではない）

ＯＳのセキュリティ強化の発想

- 前提
  - サービスを公開する以上不正なアクセスは避けられない
  - バッファオーバーフローによるプログラムの乗っ取りは完全には防げない
  - root 権限のプロセスの乗っ取りや、一般ユーザ権限プロセスからの昇格から root 権限を完全に守ることは不可能
- ならば
  - ＯＳ自身が最後の砦として、システムを守るしかない

強化の方法

- システムのリソースの利用を妥当なものかどうか可否を判断し、妥当な場合のみ利用を認める。
- 判断の基準
  - アクセス定義、あるいはアクセスポリシー
  - いわゆる「セキュリティポリシー」とはレベルが異なるので注意
- 何をもって基準を定義し、誰がいつ判断するのか？
  - 一例として、リソースにラベルをつけて、そのラベルに基づいてアクセス可否を記述させる
  - Linux の場合システムコール内で、新たな判断ロジックを追加する
- Linux への適用
  - 過剰な root 権限をうまく補うため非常に効果的
  - 制御を行うべき箇所が明確 (カーネル)

ＳＥＬｉｎｕｘが注目される理由

- かの有名な ＮＳＡ が開発、公開している
- ＮＳＡ の研究プロジェクト（ＦＬＡＳＫ）における研究成果をメインストリームのＯＳであるＬｉｎｕｘに適用（強制アクセス制御は商用 ＯＳ では実現済みだった）
- ユーザ空間や既存アプリケーションへの影響を最小にしながら強制アクセス制御を実現
  - アプリケーションから見た場合のインタフェースは同一（カーネル内部には大改造とポリシーに基づく複雑な仕組みが実装されて大変なことになっている）
  - パフォーマンスは当然低下しているが許容範囲
- 概念が難解で扱いにくい
  - 難しすぎて楽しい

ＯＳセキュリティ強化の指標

- TCSEC: Trusted Computer System Evaluation Criteria
  - 信頼されるコンピュータシステム評価の指標
  - １９８５年１２月ＤｏＤが発令 (検討は１９８１年から)
    - ＮＥＣ　ＰＣ９８０１の発表　１９８３年
    - Ｍａｃｉｎｔｏｓｈ　１９８４年
    - 一太郎の発売　１９８５年
  - 実物はこちら
  - インターネットでも参照可能。例えば、
    - http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html

ＴＣＳＥＣの目的（序文から）

- The criteria were developed with three objectives in mind:
  - (a) to provide users with a yardstick with which to assess the degree of trust that can be placed in computer systems for the secure processing of classified or other sensitive information;
  - (b) to provide guidance to manufacturers as to what to build into their new, widely-available trusted commercial products in order to satisfy trust requirements for sensitive applications; and
  - (c) to provide a basis for specifying security requirements in acquisition specifications.

TCSECとISO15408

- TCSEC=Evaluation Criteria
  - DIVISION D:
    - MINIMAL PROTECTION
  - DIVISON C
    - DISCRETIONARY PROTECTION
      - C1: Discretionary Security Protection
      - C2: Controlled Access Protection
  - DIVISION B
    - MANDATORY PROTECTION
      - B1: Labeled Security Protection
      - B2: Structured Protection
      - B3: Security Domains
  - DIVISION A
    - VERIFIED PROTECTION
      - A1: Verified Design
      - Beyond class
- ISO15408=Evaluation Assurance Level
  - EAL1:
    - Functionally Tested
  - EAL2:
    - Structurally Tested
  - EAL3:
    - Methodically Tested and Checked
  - EAL4:
    - Methodically Designed, Tested, and Reviewed
  - EAL5:
  - EAL6:
  - EAL7:

ＴＣＳＥＣ とＩＳＯ１５４０８

- ＴＣＳＥＣ における DIVISIＯＮ とＩＳＯ１５４０８ における ＥＡＬは直接対応しない（目的とスコープが異なる）。
  - 例： ＷｉｎｄｏｗｓＮＴ
    - ＴＣＳＥＣではＣ，ＩＳＯ１５４０８ではＥＡＬ４
    - 実情としては・・・
- ＴＣＳＥＣ は高信頼ＯＳの「機能要件」として、認証が終了した１９９９年以降も実質的な標準として参照されている。

LABELED SECURITY

- TCSEC における MANDATORY PROTECTION のエントリーレベル (B1)
- 上位レベルとの差分 (抜粋)
  - B2:Structured Protection
    - 明確に規定およびドキュメント化された formal security policy model
    - MACが全てのsubjectとobjectに拡張されている
    - covert channel
  - B3:Security Domains
    - reference monitor
- ラベルとアクセスポリシーに基づいた強制アクセス制御が実装されていることが、TCSEC における重要な機能要件であり、それが B1。
  - 普遍的なマイルストーン

アクセスポリシーについて

- セキュリティを強化された ＯＳ におけるアクセス可否基準を定義したもの。
- Ｌｉｎｕｘ の場合、「システムコールレベルの粒度」あるいは「システムコール単位」となる。
  - Ｌｉｎｕｘ ではシステムコールが実際の権限チェックを行うため
- 共通の文法、構文は存在せずに強制アクセス制御の実装ごとに異なる。
  - おそらく今後しばらくは変わらない
- ポリシーを保護するのは誰？
  - 規定対象である ＯＳ 自身がポリシーを保持する矛盾
  - ｒｏｏｔ とポリシー管理者の微妙な関係

SELinuxのポリシー例

- ファイルに対して可能な操作

```
#
# Define a common prefix for file access vectors.
#

common file
{
    ioctl
    read
    write
    create
    getattr
    setattr
    lock
    relabelfrom
    relabelto
    append
    unlink
    link
    rename
    execute
    swapon
    quotaon
    mounton
}
```

- システム管理者が自分のホームディレクトリを参照できるようにするための記述
  - allow sysadm_t sysadm_home_dir_t:dir { read getattr lock search ioctl add_name remove_name write };
- システム管理者のホームディレクトリに与えられているラベル

```
#
# The superuser home directory.
#
/root(/.*)?
    system_u:object_r:sysadm_home_t
/root –d system_u:object_r:sysadm_home_dir_t (-d はディレクトリのみにマッチ)
```

- システム管理者という役割に対するラベルの定義

```
#
# The following users correspond to Unix identities.
# These identities are typically assigned as the user attribute
# when login starts the user shell.  Users with access to the sysadm_r
# role should use the staff_r role instead of the user_r role when
# not in the sysadm_r.
#

user root roles { staff_r sysadm_r };
```

- ポリシーのみのダウンロード
  - http://www.coker.com.au/selinux/
  - コア開発者であるRussel氏のページ

セキュリティ強化ＯＳを理解する

- セキュリティ強化ＯＳは「セキュアなＯＳ」ではない
  - 「より細かな制御を行うことが可能な」ＯＳ
  - セキュアにするのは管理者であり利用者
- セキュリティ強化ＯＳが全ての問題や課題を解決するわけではない
  - 例えば ＤｏＳ は対象外
  - セキュリティ強化 ＯＳ にもバグは存在する（勿論！）
  - 良く聞かれる質問、「メモリ（キャッシュ）に対する攻撃にはどうするのか？」
- セキュリティ強化 ＯＳ は管理が困難
  - 適切なアクセスポリシーの管理運用
  - ＯＳ に関する深い理解とセキュリティ強化 ＯＳ の仕様の理解が前提となる
  - セキュリティ強化 ＯＳ を導入して管理運用をアウトソースするのは・・・
- Ｌｉｎｕｘにはセキュリティ強化のためのフレームワークが実装された
  - Linux Security Modules
    - カーネル内にアクセス制御を行うためのフックを登録できる（自分でカーネルを修正しなくても良い）
    - ＳＥＬｉｎｕｘ も ＬＳＭ を利用

LinuxはWindowsより安全か？

- 「セキュリティホール memo」の小島さんが NSF2003 で講演された資料
  - 「WindowsとLinuxのセキュリティ：２００３」
  - http://www.st.ryukoku.ac.jp/~kjm/security/20031024-NSF2003/Windows-linux-security-2003.pdf

- あくまで私見ですが
  - Linuxが良いと思う点
    - メールのプレビューだけで感染するようなことはない
    - chroot、User Mode Linux、読み込み専用マウント等の機能
    - 定期的にＯＳを再インストールしたりアクティベーションをしなくて良い
  - Windowsがうらやましい点
    - Windows Update に技術的な知識は不要（up2dateが同等？）
  - Linuxで困った点
    - シンボリックリンクとハードリンク
    - inode とパスの対応
  - 脆弱性が判明したとき攻撃用のプログラムは Linux のほうが書きやすいかもしれない。
  - WindowsでＩＩＳを上げるよりも Linux で Apache を上げるほうが簡単。Linux が普及したときの被害はより大規模になる恐れがある。

弊社の取り組み

- SELinuxの拡張
  - 自律的な防御機能
  - インシデントレスポンスの実装
- Linux 用アクセスポリシーの自動生成
  - アクセス履歴を採取し、それに基づきアクセスポリシーを生成する
  - Network Security Forum 2003で発表
- 強制アクセス制御によらない改ざん防止
  - ポリシーの管理運用なしに改ざんを防止する Linux
  - Linux Conference 2003で発表

- 引き続きアクセスポリシーの自動生成についてご紹介致します。