サーバーのセキュリティ対策の検証

○岡田　佳浩[1)]、千代谷一幸[1)]、早川　正人[1)]、青木　延幸[1)]、◎鬼頭　良彦[1)]
[1)] 工学研究科・工学部技術部 電子・情報技術系

はじめに

近年、名古屋大学においても不正アクセス等の報告が増えてきており、サーバーを安全に運用するためには、セキュリティ技術の向上が不可欠である。特に、サーバー管理において、セキュリティ対策における不正アクセスの防止は重要課題となっている。日常対策としてアクセス制限や暗号化通信等は広く利用されているが、それでも防ぎきれない場合も増えてきており、日常的な監視でその兆候をいかに見つけるかがポイントとなる。そこで、本研修では、OS の異なる 2 台のサーバーに、不正アクセス検知のためのセキュリティ対策ツールをインストールし、ソフトの検証と不正アクセス時のログの検証を行った。

１．検証ツール及び検証方法

セキュリティ対策ツールは多数存在する。用途別に代表的なものとして次のようなものがある。

・パスワードのチェック：John the Ripper, Crack, Kerbers, S/Key
　見破られやすいパスワードを設定しているユーザーを見つける。
・ファイルシステムの改ざんチェック：Tripwire, Trojan, Chkrootkit
　不正アクセス等によってシステムに不審なファイルが存在していないかチェックする。
・ログファイルのメッセージ監視：Snort, Swatch, Chklastlog, Chkwtmp, Logcheck
　ネットワークに流れているパケットを監視し、ログの中から不審なアクセスを見つける。
・セキュリティホールのチェック：Cops, Saint, Nessus, Tiger, Titan, Satan
　システムファイルをチェックし、パーミッション等の設定にエラーがないかチェックする。
・ポートスキャン：Nmap, PortScanner(PC)（本研修ではセキュリティツールの動作確認に使用）
　ネットワーク接続に利用されるポートの状況をチェックする。

本研修では、サーバーとなる 2 台の計算機に Solaris9、VineLinux2.1.5 の OS をインストールし、フリーなソフトであるということを基本として、下線で示したセキュリティツールについて、それぞれのサーバーにインストールし、ソフトの検証を行った。また、他のパソコンから不正アクセス等を行ったときのサーバーのログの検証を行った。その中から、外敵からのセキュリティ対策にポイントをおいた、John the Ripper、Tripwire、Snort、Swatch について報告する。

２．検証結果

2.1 Joon The Ripper

John the Ripper は、解読されやすいパスワードが使用されていないか調べるため、ホストに保存されているパスワードファイルからパスワードの解読を行う。DES、MD5 等のハッシュに対応しており、クラッシュ等に備え、ポインタ情報が一定時間ごとにセーブされる。UNIX 用の他に Windows 用もある。シャドーパスワード形式の場合、パスワードファイルとシャドーファイルを結合した解析用のパスワードファイルをアンシャドーコマンドで作成し、使用する。

John the Ripper の起動は、

./john -wordfile:password.lst ./passwdfile　　（ワードファイルモードでの起動例）

のように、John コマンドにオプションと解析対象のパスワードファイルを指定して実行する。オプションには、解析モードやハッシュ等を指定する。主な解析モード指定オプションには、

single ： ユーザーID や名前等パスワードファイルの情報を基に解析

wordfile ： 辞書ファイルを用いて解析（使用する辞書ファイルを指定する必要がある）

incremental ： 総当たり解析（全ての文字列を試す）

がある。オプションを指定しない場合、single（シングルクラックモード）、wordfile（ワードファイルモード）、incremental（インクリメンタルモード）の順に設定ファイル john.ini の記述に従い、解析を行う。

図１に John the Ripper の実行例（オプション指定なし）を示す。John the Ripper はフォアグラウンドで実行され、解読されるごとにパスワードとユーザーID が表示されていく。また、解析途中に何かキーを押すと、その時点の進行状況が表示される。test7 までの 5 つは、パスワードをユーザーID または名前と同じに設定したため、シングルクラックモードにより解読された。check10 以下はワードファイルモードにより解読された。なお、インクリメンタルモードは長時間におよぶので中止した。これらのことから、パスワードはユーザーID や名前等の内容と同一にしない。ということや、平易な単語等のパスワードの使用は避けた方がよい。ということがいえる。

```
# ./john ./passwdfile
Loaded 32 passwords with 32 different salts (FreeBSD MD5 [32/32])
check13          (check13)
test12           (check12)
kensyu           (kensyu)
test6            (test6)
test6            (test7)
guesses: 5  time: 0:00:01:20 95% (1)  c/s: 967  trying: test11931
12345678         (check10)
123abc           (test11)
monday           (check4)
blue             (test9)
friday           (check6)
red              (test8)
yellow           (test10)
network          (check1)
guesses: 13  time: 0:00:01:47 0% (2)  c/s: 976  trying: Wheels
sunday           (check5)
2222             (test2)
```

図１　John the Ripper の実行例

2.2 Tripwire

Tripwire は、ディレクトリやファイル情報の元になるデータベースを作成しておき、そのデータベースと現在のディレクトリやファイルとの整合性をチェックすることによって、ディレクトリやファイルが改竄されていないかチェックする。チェックするファイル等はポリシーファイルで指定する。また、ファイルのハッシュ値をデータベース化してチェックする。フリー版、商用版があるが、本研修では、フリーのオープンソース版を使用した。

データベースはファイルやディレクトリのハッシュ値の集合体になっており、データベースに反映させるファイル等はポリシーファイルを参照する。データベース作成でファイルが無い等の警告が出た場合、ポリシーファイルを修正し、データベースを更新する。作成したデータベースを基に定期的に整合性のチェックを行う。整合性違反があった場合、セキュリティ上問題ないか検討・対応する。また、変更があったファイルについてポリシーファイルの修正が必要なら修正する。

図２は、テキストベースのポリシーファイルから、ルールに関する部分を抜粋したものである。()で囲まれたルール属性部分と、{}で囲まれたルール設定部分との組み合わせで１つのルールが構成される。編集するのはルール設定に関する部分で、必要に応じてチェックするファイルを追加、削除またはコメントアウトする。そして、このファイルから、暗号署名を施したポリシーファイルを作成し、データベースを更新する。

```
# Tripwire Binaries
(
  rulename = "Tripwire Binaries",
  severity = $(SIG_HI)
)
{
  $(TWBIN)/siggen          -> $(SEC_BIN) ;
  $(TWBIN)/tripwire        -> $(SEC_BIN) ;
  $(TWBIN)/twadmin         -> $(SEC_BIN) ;
  $(TWBIN)/twprint         -> $(SEC_BIN) ;
# $(TWBIN)/twcopy          -> $(SEC_BIN) ;
}
```

図２　ポリシーファイル（ルール部分）

Tripwire の実行は、暗号署名を施したポリシーファイルを作成し、続いて、元となるデータベースを作成しておく。そのデータベースを基に、定期的に整合性のチェックを実施する。

図３は整合性をチェックしたときのレポートの一部である。ユーザを１人追加してチェックを行った。下線部①は出力されたレポートファイルのフルパス表示であり、下線部②は参照したデータベースのフルパス表示である。下線部③では重要な構成ファイルのグループのファイルが１つ変更されていることを示しており、このため、下線部④で整合性違反が合計で１つ存在することを示している。また、下線部⑤、⑥では、重要な構成ファイルの /etc/passwd ファイルが変更されていることを示している。なお、このとき、/etc/shadow ファイルはポリシーファイルに追加していなかったのでチェックにかからなかった。

```
Parsing policy file: /etc/tripwire/tw.pol
*** Processing Unix File System ***
Performing integrity check...
Wrote report file: /var/lib/tripwire/report/xxxx.xxxx.engg.nagoya-u.ac.jp-20041109-161615.twr
                                                                                          ①
Tripwire(R) 2.3.0 Integrity Check Report

Report generated by:       root
Report created on:         2004年11月09日 16時16分15秒
Database last updated on:  Never

===========================================================
Report Summary:
===========================================================

Host name:         xxxx.xxxx.engg.nagoya-u.ac.jp
Host IP address:   127.0.0.1
Host ID:           None
Policy file used:  /etc/tripwire/tw.pol
Configuration file used:    /etc/tripwire/tw.cfg
Database file used:         /var/lib/tripwire/xxxx.xxxx.engg.nagoya-u.ac.jp.twd
Command line used:           tripwire --check -c tw.cfg                    ②
                                          :
                                          :
                                          :
```

```
                                        :
                                        :
-------------------------------------------------------------------------
  Section: Unix File System
-------------------------------------------------------------------------

  Rule Name                 Severity Level    Added    Removed  Modified
  ---------                 --------------    -----    -------  --------
  Invariant Directories              66       0        0        0
  Temporary directories              33       0        0        0
                                        :
                                        :
  Critical system boot files        100       0        0        0
* Critical configuration files      100       0        0        1
  System boot changes               100       0        0        0      ③
                                        :
                                        :
Total objects scanned:  18040
Total violations found:  1
                          ④             :
                                        :
-------------------------------------------------------------------------
# Section: Unix File System
-------------------------------------------------------------------------


-------------------------------------------------------------------------
Rule Name: Critical configuration files (/etc/passwd)
Severity Level: 100                                    ⑤
-------------------------------------------------------------------------

Modified:
"/etc/passwd"
            ⑥                           :
                                        :
```

図３　Tripwire のレポートの出力

2.3 Snort

Snort はパターンマッチング型の NIDS（ネットワーク型侵入検知システム）で、あらかじめ用意された「ルールセット」と呼ばれる攻撃方法等をパターン化したシグネチャとマッチした場合に異常を検出する。パケットキャプチャとしても使用でき、UNIX 用の他に Windows 用もあり、オープンソースである。なお、ルールファイルは更新用プログラムを用いて cron で定期的に更新するとよい。

Snort の起動方法には、パケットキャプチャとして動作するスニファモードでの起動、検出内容をログに保存するパケットログモードでの起動、NIDS として機能するデーモンモードでの起動があるが、デーモンモードで起動し、NIDS として使用するのが一般的である。

図４に Snort が出力する監視結果のログで、アラート情報が記録されているアラートファイルの内容の一部を示す。１行目は、不正パケットの攻撃の解析結果で、CodeRed が作るバックドアを利用しようとしたアタックと思われる。２行目は[攻撃の種類]と[攻撃レベル]。３行目は、日時、送信元の MAC アドレスおよび宛先の MAC アドレス、フレームタイプ（この例では IP を示す）、パケット長と続く。４行目は、送信元の IP アドレスとポート番号および宛先の IP アドレスとポート番号、プロトコル、パケット寿命、パケットサービスタイプ、パケット ID、IP ヘッダ長、IP パケット長と続き、最後の DF はフラグメントなしを表し、パケットを分断化させないことを意味する。

```
[**] [1:1256:8] WEB-IIS CodeRed v2 root.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
11/09-16:22:32.352709 0:D0:4:xx:xx:xx -> 0:C:76:xx:xx:xx type:0x800 len:0x160
192.168.xxx.xxx:37167 -> 133.6.xxx.xxx:80 TCP TTL:61 TOS:0x0 ID:45361 IpLen:20 DgmLen:338 DF
***AP*** Seq: 0x391F3B60  Ack: 0x16438FAD  Win: 0x8218  TcpLen: 32
[Xref => http://www.cert.org/advisories/CA-2001-19.html]
```

図４　/var/log/snort/alert ファイルの内容の一部

５行目の先頭のフラグは TCP の通信の状況を示すために利用されるもので、有効なものは頭文字等の省略形で、無効なものは*で表示される。続いて、シーケンス番号、ACK 番号、ウィンドウサイズ、TCP ヘッダ長である。６行目は、参考となる URL を表示している。

Snort が出力するログは膨大な量におよび、すべてを解析するのは困難である。そこで、このログを解析するための Perl で書かれたスクリプトで SnortALog(Snort Analyser logs)というツールがある。解析の結果を、テキスト、HTML、PDF の各形式で出力でき、結果をメールで送ることもできる。図５は解析結果を HTML 形式で出力してブラウザで表示させた例である。

図５　SnortALog による HTML 形式の解析結果

2.4 Swatch

Swatch はリアルタイムでシステムログを監視し、指定した文字列にマッチするログを検知すると、BEEP 音を鳴らしたり、メールを送る等の指定したアクションを実行する不正侵入アクセス監視ツールである。Swatch のインストールには、Perl5 の他に計時、日付計算、ファイル読み込み、日付解析の４つの Perl のモジュールをあらかじめインストールしておく必要がある。

Swatch の起動は以下のように swatch コマンドと、必要に応じてオプションを指定して起動する。

```
# /usr/bin/swatch -c /etc/.swatchrc -t /var/log/messages &
```

-c オプションは設定ファイルの指定で、-t オプションは監視対象ログファイルの指定である。最後に&を付加することで、バックグラウンドで実行される。オプションなしで起動した場合、設定ファイルは起動したユーザーのホームディレクトリの.swatchrc を参照する。もし、設定ファイルが存在しなかった場合には、あらゆる行をランダムな形式で表示する。また、監視対象のログは、デフォルトでは/var/log/messages であるが、このファイルが存在しなかった場合、/var/log/syslog が監視対象となる。なお、複数のログを異なる内容で監視する場合、監視するログ個々に設定ファイルを作成し、ログごとにオプションの指定を変更して起動し、複数の Swatch を動作させる。

図６、図７にログのサンプルを示す。図６は実在するユーザーが間違ったパスワードでログインしようとした場合で、以下のような内容である。

1. ユーザ認証ができなかった。
2. User:kensyu の sshd サービスの認証に失敗した。
3. User:kensyu のパスワードが間違っている。
4. 読み取りに失敗し、接続がリセットされた。

```
# Oct 14 13:25:58 joho sshd[1143]: could not reverse map address 192.168.xxx.xxx
Oct 14 13:25:58 joho PAM_pwdb[1143]: authentication failure; (uid=0)
 -> kensyu for sshd service
Oct 14 13:25:58 joho sshd[1143]: Failed Password for kensyu from
 192.168.xxx.xxx port 3517 ssh2
Oct 14 13:26:00 joho sshd[1143]: fatal: Read from socket failed:
 Connection reset by peer
```

図６　実在するユーザーが間違ったパスワードでログインしようとした場合のログのサンプル

図７は、存在しないユーザーasdf でログインしようとした場合のログのサンプルで、以下のような内容である。

1. 不正なユーザーasdf からリクエストがあった。
2. IP アドレス：192.168.xxx.xxx を確認した。
3. 不正なユーザからのアクセスは 192.168.xxx.xxx のポート 3520 番からであった。
4. 読み取りに失敗し、接続がリセットされた。

```
# Oct 14 13:28:16 joho sshd[1144]: input_userauth_request: illegal user asdf
Oct 14 13:28:21 joho sshd[1144]: Could reserved map address 192.168.xxx.xxx
Oct 14 13:28:21 joho sshd[1144]: Failed Password for illegal user asdf
 from 192.168.xxx.xxx port 3520 ssh2
Oct 14 13:28:23 joho sshd[1143]: fatal: Read from socket failed:
 Connection reset by peer
```

図７　存在しないユーザーasdf でログインしようとした場合のログのサンプル

まとめ

セキュリティ対策ツールは用途別に様々なものがあり、サーバーを管理する上で、状況に応じてツールを組み合わせ、日常の監視を行うのが有効である。今回報告したセキュリティ対策ツールは、不審なアクセスを監視するもので、不正アクセス防止のためのアクセス制限や暗号化通信等で防ぎきれないものへの対応である。また、セキュリティ対策ツールを生かすためには、ログの判読も必要不可欠である。

参考文献

1. 高町健一郎：UNIX ネットワークセキュリティ導入ガイド，秀和システム
2. 一条博：ネットワーク監視システム，工学社
4. http://akademeia.info/main/security_lecture.htm
3. http://www.atmarkit.co.jp/index.html