snort ～システムのレーダー施設～

●awacs@hawkeye　E-mail：awacs@hawkeye.ac

侵入検知システムとは

サイトを運用する際に一番やっかいな場合は，「侵入される」ことでしょう．自サイトの被害もさることながら，他サイトにも被害を及ぼす可能性があります．しかも場合によっては「侵入された」事実が侵入者によって消去されてしまい，知らない間に被害が拡大しているかもしれません．

このようなときに，侵入検知システム（**Intrusion Detection System**：**IDS**）が役立ちます．文字どおり侵入行為を検知し，情報を提供してくれます[注1]．この情報は被害の見積もりや，復旧の際に役立ちます．

snortとは

snortとは**Martin Roesch**が開発したオープンソースの**IDS**です．特徴として，パターンマッチングによる検知を行うこと，監視対象を自由に選べること，検知に使用するパターンはテキストで記述されている，ということが挙げられます．多くの**UNIX**系**OS**と**Windows**上で動作します．なお，**snort**はパケットキャプチャとしても使用できます．

インストール

今回は**Linux**に**snort**をインストールします．

snortのインストールに際して，**libpcap**と呼ばれるライブラリのバージョン**5.0**以上（できれば最新版）をインストールしておく必要があります[注2]．インストールは難しくないので解説は割愛します．

snortの入手とインストール

snortの公式サイトへ行き最新リリースを入手します．執筆段階では**1.8.2**が最新となっています[注3]．

Red Hat系**Linux**の場合は**RPM**パッケージを利用すると便利です．いくつかのパッケージがありますが，今回は**snort-1.8.2-1snort.i386.rpm**をインストールしてください．これ以外のパッケージには，データベースによるログ管理ができるもの，**SNMP**に対応したもの，**flexresp**[注4]と呼ばれる機能が追加されているものなどがあります．

tarballを利用する場合

tarballからコンパイルする場合，上記の拡張機能を用いるには**configure**にオプションを渡す必要があります[注5]．今回はこれらの拡張機能は用いず，ログ管理は**Syslog**を主とします．この場合のインストールは標準的な方法で良く，**$./configure; make;su**して**#make install**とします．なお，デフォルトでは**/usr/local/**にインストールされます．

ルールセットの入手

snort公式サイトから最新のルールセットを入手します．**snort**の**tarball**にも収録されていますが，ルールセットは最新のものを使用するのが基本です．

これを展開すると**rules**ディレクトリが作成されます．なお，ここから先はすべて**root**で実行する必要があります．ルールセットは設定ファイルなので，**rules**ディレクトリの中身を**/etc/snort/**にコピーし，パーミッションを変更しておきましょう[注6]．

注1）筆者は個人的に，IDSは軍隊に於ける歩哨やレーダー，AWACSに相当するものであると考えています．
注2）tcpdumpやethereal などのパケットキャプチャも同様にインストールしておくと良いでしょう．昨今ではSSLを用いた通信も増えていますので，snortをSSL対応にするためにOpenSSLもインストールすることをおすすめします．
注3）公式サイトはhttp://www.snort.org/です．実は原稿締切直前に1.8.1からアップデートされました．このため本文は基本的に1.8.1ベースで書かれています．なお，非公式ではありますが，1.8.3も出ています．
注4）flexrespはパターンにマッチした通信を「強制遮断する」プロセッサです．使い方を間違えると悲劇ですが，使い方によってはおもしろい効果を上げることができるでしょう．なおRPMパッケージをインストールする場合，SSLが有効になっていません．必要であればリビルドしてください．また，snort 1.8.2ではflexrespが正常に動作しないようです．snortのcurrentのCVSツリーでは修正が行われているようなので，必要であれば，CVS，1.8.3もしくは1.8.1を使用してください．
注5）ただし，Red Hat Linux 7.xでflexrespを使用するオプションをつけた場合，コンパイルに失敗するようです．Kondara MNU/Linux2.0上でsrc.rpmをリビルドする場合は，一部specファイルを修正しましたが，問題なくコンパイルできるようです．また注4も参照してください．

snort.conf の修正

/etc/snort/snort.conf を修正します．このファイルは4ステップに分かれています．ステップに沿って修正していきます．なお行頭が「#」の場合，コメント行であることを示します．

ステップ1 ネットワークの情報

ステップ1はネットワークの情報を記述します．なおここでの書き方として「var A B」は「変数名Aは値Bである」という意味です．また「any」は文字通り「すべて」を意味します．複数の値を持たせる場合は「,」で区切り，全体を「[]」でくくることで定義可能です．ただしスペースを入れてはいけません．ここで定義された変数はルールセット中で参照されます．参照の際には変数名の前に「$」をつけます．

最初に「var HOME_NET any」という記述があり，監視対象を定義しています．ホスト単位であれば「var HOME_NET 守りたいホストのIPアドレス/32」とし，ネットワーク単位であれば「var HOME_NET ネットワークアドレス/マスク」とします[注7]．

次の「var EXTERNAL_NET any」は少々考える必要があります．EXTERNAL_NET変数は，「攻撃元となりうるIPアドレス」です．HOME_NETをanyから変更して，なおかつホストベースで運用したり，自サイトのユーザをすべて信用できるという場合は，「var EXTERNAL_NET !$HOME_NET」と記述すると良いでしょう．「!」は否定を意味します．このように定義した場合，変数EXTERNAL_NETは監視対象以外のすべてのIPアドレスである，ということになります．もし「自分のネットワーク内も信用できない」ということであれば，修正せずにanyとしておくと良いでしょう．ただし，誤検知が増大する原因の1つになります．

以降で変数SMTP，HTTP_SERVERS，SQL_SERVERS，DNS_SERVERSを定義していますが，見てのとおりサーバのアドレスの定義です．HOME_NETで定義したネットワーク内にサーバがある場合，変更は必須ではありません．他サイトにサーバがある場合，たとえばDNSのセカンダリサーバが他サイトにある場合など，きちんと定義したほうが良い場合もあります．

●表3　プリプロセッサ一覧

minfrag	細分化されたパケットを検知する
defrag	IPのデフラグメンテーションを行う
frag2	同上
stream2	TCPの再構築を行う
stream4	ステートフルインスペクション機能
stream4_reassemble	TCPの再構築を行う
http_decode	HTTPリクエストを正規化する
unidecode	ほぼ同上
rpc_decode	RPC内のエンコードを正規化する
bo	Back Orifice検知プロセッサ
telnet_decode	telnetのネゴシエーションを正規化する
portscan	portscanを検知する
portscan-ignorehosts	portscan検知から除外するホストを設定
spade	統計的手法で侵入検知を行う
arpspoof	ARPスプーフィングを検知する

ステップ2 プリプロセッサ

ステップ2はプリプロセッサの設定です．これはパケットマッチングの前処理を行うプロセッサについて設定を行います．ステップ2の基本パターンは以下のようになります．

```
preprosessor プロセッサ名 （：オプション）
```

表3にプリプロセッサの一覧を示します．必要に応じて選択してください．なお似たプリプロセッサを両方使用するのは避けたほうが良いでしょう．

●注意が必要なプリプロセッサ[注8]

注意が必要なプリプロセッサはstream4，http_decode，portscanです．

stream4は多機能な反面，過去のバージョンにはバグがあり，snortの異常終了を引き起こしていました．

http_decodeとportscanは誤検知の確率が高いで

注6）RPMパッケージをインストールした場合はすでに/etc/snortにルールセットが入っています．なお一部の方から，snortのtarballにあるルールは変では？ という指摘がありましたが，時間的都合から今回は調査を断念しました．詳細がわかり次第，章末のWebサイト（注13）で報告したいと思っています．
注7）クラスCのネットワークではマスクは24になります．DHCPでアドレスを取得している場合などはanyのままのほうが良い場合もあります．
注8）stream4は1.8.2で修正されていますが，筆者はまだ確認できていません．http_decodeは日本人が開発に参加しました．portscanは「preprocessor portscan: $HOME_NET x y portscan.log」のxを大きく，yを小さく設定してください．

●表4　出力プロセッサー覧

alert_syslog	alertをsyslogに出力する
log_tcpdump	パケットをtcpdumpフォーマットで保存する
database	alertをDBに保存する
xml	alertをxml形式で保存する
alert_unified	alertを“unified”形式で保存する
log_unified	logを“unified”形式で保存する
trap_snmp	snmpのトラップでアラートを通知する

す．**http_decode**の**UNICODE**攻撃検出機能は過去のバージョンに比べて誤検知率は低くなっているものの皆無とはいかず，そもそも対**IIS**攻撃検出用に作られているので，サイト中に**IIS**サーバが存在する場合のみ使用したほうが良いでしょう．

portscanはしきい値による判断を行っているので，誤検知ばかりという状態になる場合が多いようです．俗に言う**stealth scan**の検出にはこのしきい値を使用していないので，しきい値をできるだけ厳しくとるほうが良いでしょう．

snort運用ホストの能力が低い場合，**CPU**を酷使する**frag2**や**stream**系プロセッサなども止めておいたほうが良いかもしれません．ただしフラグメント攻撃を受けた場合，検知できない場合があります．

ステップ3 出力

ステップ**3**は出力を司るプロセッサについての設定です．こちらの書式は以下のようになります．

```
output プロセッサ名（:オプション）
```

ログ形式として表**4**に示す形式が使用できます．一般的には**syslog**出力が便利ですから，**alert_syslog**だけコメントを解除しておけば良いでしょう．

ステップ4 ルールセット

ステップ**4**では，**snort**の命とも言えるルールセットの設定を行います．ルールセットは対象によって別ファイルで定義し，インクルードします．最初はすべてをインクルードして実行してみると良いでしょう．

実行

できるだけ**snort**専用のユーザアカウントを作り

●図3　snortの実行結果

```
Log directory = /var/log/snort
  (途中省略)
-*> Snort! <*-
Version 1.8.2 (Build 86)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)


==========================================
==========
Snort analyzed 2 out of 2 packets, dropping 0(0.000%)
packets

Breakdown by protocol:                Action Stats:
    TCP: 0          (0.000%)          ALERTS: 2
    UDP: 0          (0.000%)          LOGGED: 2
   ICMP: 2          (100.000%)         PASSED: 0
    ARP: 0          (0.000%)
   IPv6: 0          (0.000%)
    IPX: 0          (0.000%)
  OTHER: 0          (0.000%)
DISCARD: 0          (0.000%)
  (以下省略)
```

ログイン不可能にしておき，そのアカウントで**snort**を実行するようにしておきましょう．仮に**snort**が乗っ取られたとしても，システム権限まで取られることは防げます．

まずはフォアグラウンドで確認

上記の設定をすべて行えば（最低ステップ**1**を設定すれば）**snort**はいつでも実行可能状態になります．

以下のように実行するとフォアグラウンドプロセスとして起動し，**lo**インターフェースを監視します．もしここですぐ終了してしまう場合は，**snort.conf**のどこかに間違った記述があるのでよく確認しましょう．

```
# snort -i lo -c /etc/snort/snort.conf
```

起動できたらルールに引っかかりそうな行為を行い反応を見てみましょう．先の設定でルールセットをすべて選択している場合，**127.0.0.1**に**ping**を打つことで確認できると思います．この後**snort**を Ctrl+C で終了してください．実行結果のサマリが出力され，**ICMP**を補足し，**Alerts**を出力した旨のメッセージが表示されると思います（図**3**）．

デーモンとして実行

筆者の場合，以下のようにして実行しています．

```
# snort -i eth0 -D -u snort -g snort -l
/var/log/snort -c /etc/snort/snort.conf
```

（実際は1行）

これは，ユーザ**snort**，グループ**snort**で，**/etc/snort/snort.conf**を設定ファイルとして使用し，**eth0**をデーモンモードで監視，パケットダンプを**/var/log/snort**へ出力する，というものです．必要に応じてオプションを付加しましょう．ただし，**-s**オプションをつけた場合，設定したポストプロセッサが無効になる上，**-l**オプションも無効になるようなので注意してください．

また，実際に運用するときはネットワークが起動した直後に実行されるようにしましょう．**/etc/rc.d**などに直接記述してもかまわないと思います．

ルールメンテナンス

すべてのルールで実行して**1**週間程度運用してみてください．大量のうんざりするようなログが出力されていると思います．これが「あなたのネットワークの実態」です．そのほとんどが無害なものでしょう，きっと….

ルールセットの調整

ここからが本当に**snort**を使いこなすために必要な作業です．まず，その大量のログを確認し，問題ないものであればそれを捕捉するルールを削除します．特定のホストが出力しているだけであれば，様子見としてそのホストだけ監視対象から外す，といったルールに書き換えても良いでしょう．監視対象とは関係のないルールは削ります．そしてログがほとんど出なくなったとき，ルールセットが自分のサイトに合ったものになります．

●ログ閲覧ツール[注9]

ログを読みやすくするツールがいくつも公開されているので，それを利用すると良いでしょう．筆者は**ACID**という，データベース化された**snort**ログを**Web**ベースで確認できるツールを使っています．作者が外国人なので質問しづらいということであれば，**smoker**氏の作成したログを**HTML**に整形する**Perl**スクリプトを利用するのも良いでしょう．彼は日本人なので，要望を伝えるのも日本語で構わないぶん楽です．

●表5　rule_option一覧

msg	logやalertで表示する文字を指定
logto	logを書き込むファイル名を指定
ttl	IPヘッダのTTL値を評価
tos	IPヘッダのTOS値を評価
id	IPヘッダのフラグメントID値を評価
ipoption	IPオプションビットを評価
fragbits	IPヘッダのフラグメントビットを評価
dsize	パケットペイロードのサイズを評価
flags	TCPフラッグを評価
seq	TCPシーケンスナンバを評価
ack	TCP-ACKフィールドを評価
itype	ICMPのタイプを評価
icode	ICMPのコードを評価
icmp_id	ICMP_ECHO_IDフィールドを評価
icmp_seq	ICMP_ECHOのシーケンスナンバを評価
content	パケットペイロード中のパターンを探す
content-list	ペイロード中の複数パターンを探す
offset	何バイト目からパターンを探しはじめるかを指定
depth	何バイト目までパターンを探すかを指定
nocase	大文字小文字の区別をしない
session	セッションのアプリケーションレイヤの情報を出力
rpc	指定されたRPCサービスを見る
resp	レスポンスを指定（flexrespの機能）
react	アクションを指定（flexrespの機能）
reference	リファレンス情報を指定
sid	snortのrule-idを指定
rev	ruleのリビジョンを指定
classtype	ruleのクラス分けを指定
priority	ruleの重要度を指定
uricontent	URIのパターンを指定
tag	ルールにマッチした場合にタグをつける
ip_proto	IPのプロトコル値を指定
sameip	ソースとディスティネーションが同じかどうか評価
stateless	ストリームの状態にかかわらず妥当と判断
regex	ワイルドカードを利用したパターンマッチングを行う

ルールに関する注意

ルールセットはひんぱんに更新されます．古いルールセットでは新しいタイプの攻撃を捕捉できない場合が多々あります．このため新しいルールセットが公開されていないかどうか確認することを日常業務としてください[注10]．

場合によっては自分でルールを作る必要が生じる場合もあるでしょう．ルールの書式自体はシンプルですが，難しいのは，アドレス，プロトコル，ポート，パケット内容，これらすべてについて理解している必要があります．**1**からルールを書くのは簡単ではありません．なお，ページの都合上，今回は簡単な図表でルールの書式を説明するにと

注9）多くのツールがtarballのcontribディレクトリに入っています．smoker氏のスクリプトはhttp://smokerz.zone.ne.jp/のアーカイブから取得できます．
注10）ルールセットは公式サイトのDevelopmentにあります．また，http://www.whitehats.com/も情報源としてチェックするとよいでしょう．

●図4　ルールの記述フォーマット

```
  alert     tcp
   log      udp
{  pass  } {icmp} ip_address port_number {->} ip_address port_number (rule_option;rule_option;…)
 activate    ip                          <>    dynamic
```

ip_addressはCIDR形式で記述．カンマ区切りで複数指定可．この時，[]でくくること．先頭に!をつけた場合，ip_address以外，という指定になる．port_numberはカンマ区切りで複数指定可．また，：は範囲指定．

どめさせていただきました[注11]（図4，表5）．

運用について

サイト防御のためのツールは，結局使う人の知恵次第です．どんなツールを使っても，攻撃者にそれを迂回されては意味がありません．攻撃者との知恵比べにつねに勝ち続けなければならないのです．

ログは転送する

snortを実行しているホストのログは，できるかぎりほかのホストに転送しましょう．仮にこのサイトに侵入して足跡を消したいと思う人がいても，**snort**のログが送られているサイトも攻略しなければならなくなります．ただし大量のログがたまることになるため，ディスクフルによる**DoS**攻撃が成立しないよう，できるかぎり大きなディスクを持つ必要があります．

snort運用ホストは要塞化する

IDSはレーダーなので，存在がわかった時点で最重要ターゲットになります．アフガンでも最初に攻撃を受けたのはレーダー施設でした．同様のことがネットワークでもいえます．**snort**運用ホストは攻撃されることがないよう，要塞化を行っておきましょう．

IDSの誤検知

ルールセットのメンテナンスを困難にする要因の**1**つとして，**IDS**の誤検知があります．誤検知には攻撃でないものを攻撃と間違える**false positive**と，攻撃を見逃してしまう**false negative**があります．

前者はルールセットが厳密でない場合によく生じます[注12]．**snort**の出力したパケットダンプの内容を見て，誤検知かどうか確認する作業が必要です．また，ルールをもう少し厳密なものに修正しておくと良いでしょう．必要な場合は（承認を取って）パケットキャプチャで人間が監視する，ということも考慮に入れてください．監視対象に関係のないルールで誤検知している場合は何も考えず削除します．

後者はルールセットが古かったり攻撃パターンが既知のものと違う場合に生じます．現状のルールが本当に正しいかということを，いつも考えておかなければなりません．必要に応じて，ルールセットのバージョンアップや修正を行いましょう．

snortは見張るだけ

最後に何よりも重要なことを記します．まず**snort**は単なる見張り番であって，番犬ではないということです．なにも守ってはくれません．このため**1**番最初にやるべきことは**snort**のインストールではなく，ホストをきっちり設定することです．ファイアウォールや**IDS**があるから安全，というのは単なる誤解であって，それ以上の何者でもない，ということを理解してください．

おわりに

駆け足で**snort**について記しましたが，残念なことにこれだけ奥の深いプロダクトをすべて解説することはできませんでした．説明不足などありましたら，筆者までご連絡いただけると幸いです[注13]．

最後になりますが，本稿を執筆する機会を与えてくださったすべての方と，最後まで目を通してくださった読者の方に感謝します．SD

注11）ルール記述の詳細は，docディレクトリ内にPDF化された英語のドキュメントがありますので，こちらを参照してください．
注12）ありがちなのは，ポート番号のみで判別しているようなルールを使用している場合です．たまたまFTPなどでそのポートを使用した場合にも，誤検知してしまいます．
注13）「Micky's security institute（http://www.hawkeye.ac/micky/）」や「しかPの単なるメモ帳（http://www.yk.rim.or.jp/~shikap/）」にsnortの情報があります．今回の記事についてのフォローは「しかPの単なるメモ帳」の一部を間借りして記載しておきますのでご確認ください．