学部サーバへの IDS の導入・運用

青木 敏裕

熊本大学工学部技術部　電気情報技術系

1. はじめに

熊本大学工学部技術部では当大学工学部のサーバ群（DNS,Web 他）を管理している。これまでセキュリティ面に関しては、主に各マシンでファイアウォールを設定し運用してきた。しかし近年複雑化する攻撃に加え、外部からの攻撃に留まらず学内のウィルス感染 PC からの攻撃も増加しており、これらの攻撃を検知する為に侵入検知システム（以下 IDS）の導入を検討した。当初は商用の IDS の導入を検討したが、予算の関係上導入は難しいという結論に至った。そこでオープンソースの IDS である「Snort」を利用して IDS を構築することになった。本発表では Snort の導入・初期の運用時の問題について報告を行なう。

2. Snort　の概要

Snort はソースファイア社により開発されているオープンソースの IDS である。IDS のタイプとしてはネットワーク型・シグネチャ型に分類される物である。基本アーキテクチャとしてスニファ、プリプロセッサ、検知エンジン、出力の 4 つのコンポーネントから構成されている。このうち検知エンジンが、取得したパケットに対してシグネチャに従いパターンマッチを行なう。図 1 は作成したシグネチャの例である。

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"SSH Login(SYN packet)";
flow:from_client; flags:S,12; sid:1000000; rev:1;)
```

図 1　シグネチャの記述例

上記のようにパケットのプロトコル（tcp,udp,icmp,ip）、パケットがどちら向きに流れてきたか、送受信のポートなどを記述する。図 1 のシグネチャの場合、外部から内部ネットワークへの SSH のパケットが alert として出力される。

3. システム構成

今回は Snort を Linux 上にインストールし、図 2 のような構成で導入した。L2 スイッチにポートミラーリング機能のある物を用い、スイッチ配下のサーバ群を監視している。

動作環境は以下の通りである。

- OS：Ubuntu server 10.4LTS（64bit）
- Snort：2.8.5.2
- CPU：Intel Xeon X3363（2.83GHz）
- Memory：8GB

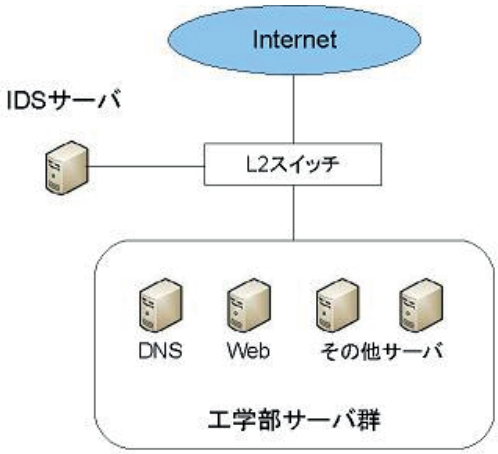


図 2　システム概略図

4. 連携ツール

Snort は単体でも利用できるが、管理の手間を軽減するツールが多数公開されている。今回は以下の 2 種のツールを利用している。

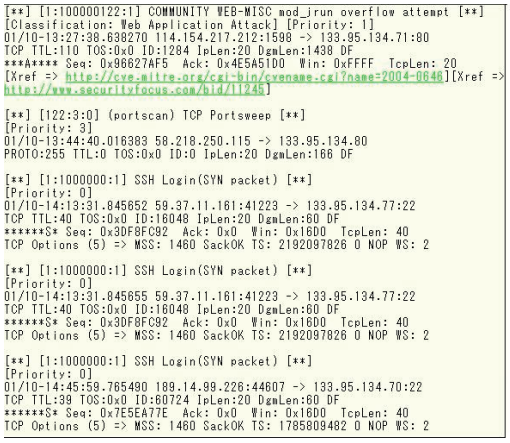
・Oinkmaster

Snort のルールファイルは不定期にアップデートされている。Oinkmaster はこれを自動的にアップデートするツールである。conf ファイルに記述することで、特定のシグネチャを自動的にコメントアウトする機能も備えている。

・SnortSnarf

SnortSnarf は Snort の出力したアラートを見やすい形で HTML に整形して出力するツールである。（図 3）

Alert の種別、発信元/先ごとに表示することが出来る。



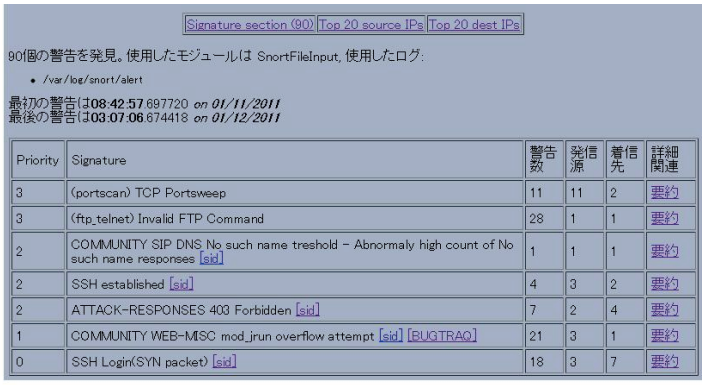
```
[**] [1:100000122:1] COMMUNITY WEB-MISC mod_jrun overflow attempt [**]
[Classification: Web Application Attack] [Priority: 1]
01/10-13:27:38.638270 114.154.217.212:1598 -> 133.95.134.71:80
TCP TTL:110 TOS:0x0 ID:1284 IpLen:20 DgmLen:1438 DF
***A**** Seq: 0x86627AF5  Ack: 0x4E5A51D0  Win: 0xFFFF  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2004-0646][Xref =>
http://www.securityfocus.com/bid/11245]

[**] [122:3:0] (portscan) TCP Portsweep [**]
[Priority: 3]
01/10-13:44:40.016383 58.218.250.115 -> 133.95.134.80
PROTO:255 TTL:0 TOS:0x0 ID:0 IpLen:20 DgmLen:166 DF

[**] [1:1000000:1] SSH Login(SYN packet) [**]
[Priority: 0]
01/10-14:13:31.845652 59.37.11.161:41223 -> 133.95.134.77:22
TCP TTL:40 TOS:0x0 ID:16048 IpLen:20 DgmLen:60 DF
******S* Seq: 0x3DF8FC92  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 2192097826 0 NOP WS: 2

[**] [1:1000000:1] SSH Login(SYN packet) [**]
[Priority: 0]
01/10-14:13:31.845655 59.37.11.161:41223 -> 133.95.134.77:22
TCP TTL:40 TOS:0x0 ID:16048 IpLen:20 DgmLen:60 DF
******S* Seq: 0x3DF8FC92  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 2192097826 0 NOP WS: 2

[**] [1:1000000:1] SSH Login(SYN packet) [**]
[Priority: 0]
01/10-14:45:59.765490 189.14.99.226:44607 -> 133.95.134.70:22
TCP TTL:39 TOS:0x0 ID:60724 IpLen:20 DgmLen:60 DF
******S* Seq: 0x7E5EA77E  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 1785808482 0 NOP WS: 2
```



Signature section (90) | Top 20 source IPs | Top 20 dest IPs

90個の警告を発見。使用したモジュールは SnortFileInput, 使用したログ:

- /var/log/snort/alert

最初の警告は08:42:57.697720 *on 01/11/2011*
最後の警告は03:07:06.674418 *on 01/12/2011*

Priority	Signature	警告数	発信源	着信先	詳細関連
3	(portscan) TCP Portsweep	11	11	2	要約
3	(ftp_telnet) Invalid FTP Command	28	1	1	要約
2	COMMUNITY SIP DNS No such name treshold - Abnormaly high count of No such name responses [sid]	1	1	1	要約
2	SSH established [sid]	4	3	2	要約
2	ATTACK-RESPONSES 403 Forbidden [sid]	7	2	4	要約
1	COMMUNITY WEB-MISC mod_jrun overflow attempt [sid] [BUGTRAQ]	21	3	1	要約
0	SSH Login(SYN packet) [sid]	18	3	7	要約

図 3　素の alert ログと SnortSnarf による整形後の alert の HTML

5.　実際の運用

実際に運用を始めたところ、以下のような成果があがった。

- 数日後に大量のアラートが出力され、サーバの 1 つが現在進行形で攻撃を受けていることが判明（Ping を数千/日で送りつけていたホストがあった）
- 学内のウィルス感染 PC からの攻撃（ポートスキャン、SSH ブルートフォース）の検知
- ファイアウォールの記述ミスで正しくパケットを破棄していなかったことが判明

成果のあった一方、運用直後は大量の誤検知に悩まされることとなった。誤検知は常に IDS に付きまとう問題である。これに対しては、誤検知の多いシグネチャを無効にする（今回はクロウラと TCP/IP flood に関するシグネチャを無効化）等の対策が考えられるが、最終的に False Positive と False Negative のバランスになってしまうと思われる。基本的には前者が多い分には攻撃も確実に検出できるため良いが、あまりに誤検出が多い場合「本物の警告」を見逃してしまいかねない。もちろん後者が多い場合、攻撃そのものの見逃しが多くなってしまうだろう。

また、ルールアップデートにより Snort が動作しない事態も起きた。配布されていたルールファイルに不備があったようで、フォーラムを参考に該当シグネチャを無効化することによって対応した。

6.　まとめ

今回オープンソースである Snort を用いて IDS を構築し、攻撃を検知するなど成果があがった。一方で誤検知の問題に遭遇したことで、IDS は構築した後こそが手間がかかるということも実感した。

今後の課題として日々の傾向をグラフ化し、より見易い形に整形すること、異常が多い場合にメール等での自動連絡が出来るような設定などを試してみたい。

7.　参考 URL

[1] @IT　“Snortでつくる侵入検知システム”　http://www.atmarkit.co.jp/fsecurity/rensai/snort01/snort01.html

[2] Linuxサーバ構築ガイド　http://safe-linux.homeip.net/security/linux-snort3-07.html