7P-3

パケットフローデータベースとネットワーク侵入検知

白木原　祐介[†]　金子　邦彦[††]　牧之内　顕文[††]
[†]九州大学 電気情報工学科
[††]九州大学 大学院システム情報科学研究院

IP，ICMP パケットについて，ネットワークデバイスに到達するすべてのパケットをキャプチャし，パケットのヘッダとペイロードの先頭部分をリレーショナルデータベース HiRDB に格納し，その上で各種の分析を行うシステムであるパケットフローデータベースを作成した．パケットフローデータベースには，Ethernet ヘッダ，Options の一部を含む IP ヘッダ，ICMP ヘッダ，TCP/UDP ヘッダ，TCP/UDP ペイロードの先頭最大 64 バイト，HTTP リクエストメッセージの 4 つのヘッダフィールドが格納される．HiRDB の SQL プロセッサを使い，パケットフローデータベース上で SQL での集約問い合わせを容易に行える．データ分析による攻撃的侵入検知について報告する．

1　はじめに

フリーソフトの侵入検知システム Snort[1]は既知の侵入についてのシグネチャ辞書として持ち，全パケットをキャプチャし，シグネチャ辞書とのマッチングによって侵入検知を行う．筆者らの研究室（九州大学内）の LAN（20 台程度の機器がつながった LAN であり，Web サーバがある）において，試しに Snort v2.4 を，2005 年 12 月 16 日 9 時 37 分 38 秒から 19 日 10 時 13 分 22 秒の間稼動させた結果，50 種類のアラートメッセージが出た．図 1 に上位 10 個を示す．アラートメッセージを解析した結果，攻撃はすべて未遂に終わっていることが分かった．

Sid	message	count
1411	SNMP public access udp	1641
1417	SNMP request udp	1641
1201	ATTACK-RESPONSES 403 Forbidden	442
590	RPC portmap ypserv request UDP	423
1852	WEB-MISC robots.txt access	320
1923	RPC portmap proxy attempt UDP	290
1321	BAD-TRAFFIC 0 ttl	290
2003	MS-SQL Worm propagation attempt	196
2004	MS-SQL Worm propagation attempt OUTBOUND	196
2050	MS-SQL version overflow attempt	196

図 1　Snort v2.4 を 2005 年 12 月 16 日 9 時 37 分 38 秒から 19 日 10 時 13 分 22 秒の間稼動させた結果出たアラートメッセージのうち上位 10 個

Implementation of a Packet Database using an Relational Database System and Intrusion Detection by Aggregate Queries
[†]Department of Electrical Engineering & Computer Science, Kyushu University
[††]Graduate School of Information Science and Electrical Engineering, Kyushu University

以上のようにシグネチャ辞書型の侵入検知システムは既知の攻撃的侵入検知（未遂も含め）に有効である．一方，アラートメッセージの解析において，全パケットの通信記録が役立つ．さらには，ネットワークのパケットトラフィックの分析のためにも役立つ．パケットフローデータベースの実装を本稿で報告する．

2　パケットフローデータベースキャプチャプログラム

パケットフローデータベースは，ネットワークデバイスに到達する IP，ICMP パケットをキャプチャしてパケットのヘッダの各フィールドとペイロード先頭部分を得るキャプチャプログラムを作り，データベースを構築をする．

パケットをキャプチャするために，フリーソフトウエアのパケットキャプチャライブラリ Libpcap[2]を使用した．Libpcap をプロミスキャスモードで動作させ，「IP」という BPF(BSD Packet Filter)を使うように設定すると，Libpcap を動作させている計算機のネットワークデバイスに到達するすべての IP 及び ICMP パケットをキャプチャする（使用した OS は FreeBSD バージョン 4）．さらに，タイムスタンプ及び元のフレームのバイト数などの情報を付加する機能も持つ．FCS を除く Ethernet フレーム全体を読み込めるように，Libpcap においてキャプチャする最大バイト数を 1514 バイトに設定して動作させている．

キャプチャプログラムは，Libpcap を使い，ヘッダの各フィールド値とペイロード先頭部分を得る．パケットフローデータベースには，各パケットの Ethernet，IP，TCP，UDP，ICMP のヘッダ値全てと，TCP/UDP ペイロードの先頭部分を格納している．また，現在の実装では，ペイロード全部は格納していないし，TCP の Options，ICMP の Data も格納していない．これらが，トラフィックの分析には不要と判断したためだが，これら全部を格納するように書き換えることは容易に可能である．

Ethernet ヘッダについては，Destination Ethernet Address，Source Ethernet Address，Type の 3 つのフィールドを格納する．IP ヘッダについては，全フィールド（RFC791 参照）を格納する．但し，IP の Options については，ソースルーティングに関係して（Loose Record and Source Route あるいは Strict Record and Source Route）Type，Length，Pointer，Route という 4 つのフィールドが定まっているので，この 4 つを格納する．TCP ヘッダ（RFC793 と RFC3168 参照）については，Options と Reserved を除く全フィールドを格納する．

```
Dec 17 04:55:10 cap sshd[24786]: refused connect from 222.239.220.118
(222.239.220.118)
Dec 17 11:06:19 cap sshd[25084]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 17 12:03:53 cap sshd[25129]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 17 17:24:37 cap sshd[25381]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 17 18:12:20 cap sshd[25420]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 17 19:46:20 cap sshd[25493]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 17 19:48:58 cap sshd[25494]: refused connect from 203.52.90.140 (203.52.90.140)
Dec 17 20:01:01 cap sshd[25510]: refused connect from 203.52.90.140 (203.52.90.140)
Dec 17 20:06:55 cap sshd[25514]: refused connect from 207-234-129-
112.ptr.primarydns.com (207.234.129.112)
Dec 19 00:55:42 cap sshd[27184]: warning: /etc/hosts.allow, line 1: can't verify
hostname: getaddrinfo(alpha.shiftpoint.net, AF_INET) failed
Dec 19 00:55:42 cap sshd[27184]: refused connect from 209.152.169.217
(209.152.169.217)
Dec 19 06:00:22 cap sshd[27727]: refused connect from 221.11.140.231
(221.11.140.231)
Dec 19 06:01:09 cap sshd[27728]: refused connect from 211.154.45.131
(211.154.45.131)
```

図 2 auth.log に記録されていた ssh における接続拒否

但し，ECN（RFC3168 参照）は有効な値が入っているときのみ格納する．UDP ヘッダ（RFC768 参照）は，全フィールドを格納する．ICMP ヘッダについては Type フィールドの値によってフォーマットが変わってくる．そこで RFC792 で定義されている全メッセージタイプの，Data を除く全フィールドを格納するように実装している．

3 パケットフローデータベース

Snort v2.4 のアラートメッセージに含まれるヘッダ項目は，それぞれ数値あるいは文字列のいずれかのタイプ（テキスト形式で表現）になっているので，パケットフローデータベースにおいては，ヘッダ項目は，Snort v2.4 のアラートメッセージでのタイプと同じタイプで格納している．

Snort のアラートメッセージには無いが，パケットフローデータベースにのみ含まれるヘッダ項目としては，IP ヘッダにおける Type Of Service，Fragment Offset，Header Checksum，TCP ヘッダにおける Checksum，Urgent Pointer，UDP ヘッダにおけるの Checksum，ICMP ヘッダにおける Checksum，Pointer，Gateway Internet Address，Originate Timestamp，Receive Timestamp，Transmit Timestamp がある．

パケットフローデータベースの実装では，商用リレーショナルデータベースである株式会社 日立製作所製 HiRDB バージョン 7 を使用した．パケットデータを格納するために 58 の属性を持つリレーション Packet を定義した．

4 集約問い合わせの例

図 1 と同じ 2005 年 12 月 16 日 9 時 37 分 38 秒から 19 日 10 時 13 分 22 秒までの間システムを動作させ，5769745 個のパケットからなるパケットフローデータベースを構築した．以下，このデータベース上で HiRDB の SQL プロセッサを使い，パケットヘッダが持つ情報について SQL での集約問い合わせによるデータ分析の結果を報告する．

SQL の集約問い合わせで，SSH によるパスワード総当り攻撃の検知を試みた．SSH はリモートアクセスで他のマシンにログインしさまざまなコマンドを実行するためのプログラムである [3]．パケットフローデータベースを動かしているマシンの auth.log ファイルを見ると図 1 のように動作時間内にいくつかのアドレスから ssh での 15 回の接続拒否が起こっていた．実際に攻撃を受けていることがわかる．

SQL

```
select destinationaddress,count(*)
from packet
where
   sourceaddress = '133.5.18.174'
and sourceport = 22
group by destinationaddress;
```

結果

DESTINATIONADDRESS	COUNT(*)
133.5.18.186	6040
222.239.220.118	4
207.234.129.112	18
203.52.90.140	6
209.152.169.217	3
221.11.140.231	6
211.154.45.131	6

SQL

```
select destinationaddress,count(*)
from packet2
where
   sourceaddress = '133.5.18.174'
and sourceport = 22
and fin = '1'
group by destinationaddress;
```

結果

DESTINATIONADDRESS	COUNT(*)
133.5.18.186	1
222.239.220.118	1
207.234.129.112	6
203.52.90.140	2
209.152.169.217	1
221.11.140.231	2
211.154.45.131	2

図 3 ssh サーバからクライアントへの Destination Address ごとの全パケット数を求める SQL と結果（左）
そのうち FIN フラグが 1 のパケット数を求める SQL と結果（右）

SSH の通信でペイロード部はすべて暗号化されているので Snort v2.4 で検知することはできない．しかしこの攻撃は短い TCP セッションが繰り返されるという特徴があるので，ssh サーバからクライアントへの全パケット数と，そのうち FIN フラグが 1 であるパケット数との比率を調べる．これをもとに SQL でクライアントへの全パケット数とそのうち FIN フラグが 1 であるパケット数を求めるための SQL での集約問い合わせと結果が図 2 と図 3 になる．

この結果から，サーバからの正規クライアントへの全パケット数と比べて FIN フラグが 1 であるパケットはとても少ない．しかし，攻撃的侵入においてはクライアントへのパケット数全体に対して，図 3 では 30%近くも FIN フラグが 1 であるパケットが発生している．これは攻撃時の特徴といえる．

5 おわりに

集約問い合わせによるデータ分析により，Snort では検知できない攻撃的侵入を検知することができた．

謝辞

本論文作成にあたり，「日立 HiRDB アカデミック支援プログラム」に参加させていただいた株式会社 日立製作所様に感謝いたします．

参考文献

[1] http://www.snort.org/
[2] http://www.tcpdump.org/
[3] http://www.snailbook.com/docs/protocol-1.5.txt