sFlow によるネットワークトラフィック監視システムの提案

斎田　佳輝[*]　　市川　本浩[*]　　中村　豊[**]　　藤川　和利[*]　　砂原　秀樹[*]
*奈良先端科学技術大学院大学　　**九州工業大学

Network Analysis System using sFlow

Yoshiaki Saita[*]　Motohiro Ichikawa　Yutaka Nakamura[**]　Kazutoshi Fujikawa[*]　Hideki Sunahara[*]
*Nara Institute of Science and Technology　　**Kyushu Institute of Technology

1. はじめに

コンピュータネットワークの広帯域化にともない，ネットワーク上で利用されているアプリケーションの多様化が進んでいる．これにともない，健全で安全なネットワーク環境を提供する必要性が増しつつある．健全で安全なネットワークを維持するためには P2P ファイル共有ソフトによる違法性の高いコンテンツ転送の発見，ウイルスによる被害拡大の防止など要求が多様化している．ネットワーク運用者がこれらの通信を検出するには，ネットワーク型 Intrusion Detection System (IDS)を用いるのが一般的である．

ネットワーク型 IDS は通常，すべてのパケットを取得し，解析することによりネットワーク管理者が意図しないパケットかどうかを判断する．しかしネットワークの広帯域化，多様化によりネットワーク上を流れるパケット一つ一つを解析する事は困難になりつつある．こうした背景の中，各ネットワーク機器ベンダにおいては様々な角度からトラフィックを観測できる技術を提案，提供している．本学においては，2003 年 5 月より Cisco Systems 社が提案する NetFlow[2]を利用したネットワークトラフィック監視システムを構築し，実際に運用している[8]．さらに 2005 年 3 月より，InMon 社が提案する sFlow[5]を利用した計測を開始した．本稿では，sFlow を利用したネットワーク監視システムについて述べる．

以下 2 章では既存技術の問題点について述べる．3 章では我々が現在運用している sFlow を利用したネットワーク監視システムの設計について述べる．4 章では，我々の研究グループで開発した NetFlow を利用したネットワーク監視システムへの適応について述べる．5 章では NetFlow を利用したネットワーク監視システムの問題点とその解決に向けた基礎実験について述べる．

2. 既存技術

本章では，既存のネットワークトラフィック計測の技術を紹介する．

2.1 MRTG

Multi Router Traffic Grapher (MRTG) [2]はルータや計測機器より取れるデータを可視化するツールである．長期にわたり計測が可能であり，記録するデータを集約するため，保存に必要な領域も少ない．また，簡単に導入できるため，広く利用されている．しかしながら，過去のデータほどその粒度は荒く，詳細なデータの分析は困難である．

2.2 tcpdump

tcpdump[7]は一般的なコンピュータで動作するネットワーク計測ツールである．導入されたコンピュータのネットワークインタフェースに到達するパケットを表示・記録保存できる．すべてのパケットを記録するため，保存されるデータ容量が大きくなる．特に，近年の数 Gbps オーダーから数十 Gbps オーダーのネットワークリンクの計測に対しては，パケットの取りこぼしなどが発生する場合もある．また，計測時間にあわせたデータ保存領域が必要になり，長期的な計測に用いるには現実的に難しい．

2.3 NetFlow

NetFlow は Cisco Systems 社が提案するサンプリングベースのネットワークトラフィック計測技術である．同社のルータや各ベンダからも対応機器が販売されており，それらを導入すれば利用できる．ルータに実装されているため，ネットワークの基幹部分での計測が可能である．NetFlow の情報は，ルータのルーティング用のキャッシュから生成している．このため機器への負荷を低く抑えている．また，NetFlow ではトラフィック情報を TCP のコネクションの概念に近い独自の“フロー”と呼ばれるデータに集約し，トラフィック情報を報告する．広帯域に対応可能である反面，取得できるデータが Layer 2 から Layer 4 までのヘッダ情報の一部に限られるため，トラフィックの解析には工夫が必要である．

2.4 sFlow

sFlow は InMon 社が提案するトラフィック計測技術である．Foundry Networks 社やアラクサラネットワークス社のルータなどに実装されている．sFlow の情報は，ルータを通過するパケットを，あらかじめ設定されたサンプリングレートで観測する．観測されたパケットはルータで変更されることなく記録される．つまり，観測されたパケットデータは実質 tcpdump のデータと同等のものといえる．観測されたパケットは，sFlow データグラムというパケットに格納されルータから送信される．sFlow データグラムには，sFlow のヘッダ，および，サンプリングされたパケットの上位 256byte が格納されている．また sFlow データグラムには，上記のデータ以外にも，Simple Network Management Protocol (SNMP)を利用したときに取得できるルータの各統計情報も送信するため，SNMP エージェントを利用せずに機器の実トラフィック量などを知ることもできる．

NetFlow では，各 Layer のヘッダ情報の一部しか取得することができない．そのため，その通信内容を含めたデータ解析ができなかった．しかし，sFlow ではルータを通過するパケットをそのまま記録するため，既存の解析手法に適応するなどの汎用性が期待できる．

3. sFlow を利用したネットワーク監視システムの設計と構築

本章では，sFlow を利用したネットワーク監視のシステムの設計に関して述べる．

3.1 用語定義

本節では，本稿で用いる用語の定義について述べる．sFlow を利用したトラフィック計測ができるルータを，エージェントと呼ぶ．エージェントが送信する sFlow に関するパケットを sFlow データグラムと呼ぶ．エージェントからの sFlow データグラムを受信し，解析･蓄積する機器をコレクタと呼ぶ．

3.2 収集データ

本システムにおいて，我々はエージェントが送信する sFlow データグラムのうち，観測パケット情報が含まれるデータグラムを取得・解析し，データベースへ保存する．機器情報が含まれる sFlow データグラムは収集対象としていない．収集される sFlow データグラムのフォーマットを図 1 に示す．また，各項目の説明を表 1 に示す．

sFlow データグラムのバージョンは 2，4，5 が存在する．本システム上で利用されているエージェントが送信する sFlow データグラムのバージョンは 2 および 4 である．収集対象としている sFlow データグラムでの両者の差は無いため，両バージョンとも問題なく収集できる．

我々は，sFlow データグラムの各項目をデコードし，データベースへ蓄積するためのソフトウェアを開発した．sFlow データグラムにはデータグラム中に含まれる観測パケットが実際観測された時刻は記録されていない．そのため本ソフトウェアでは，コレクタが sFlow データグラムを受信した時刻の記録を行う．その時刻を，記録されているパケットが発生した時刻とし，データベースへ保存する．この時，実際のパケットがルータを通過した時刻と，データベースに記録される時刻には若干のずれが生じる．しかし，そのずれはたかだか 1 秒未満である．

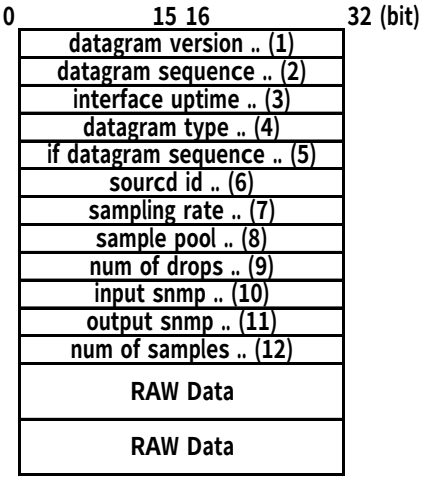

図 1 sFlow データグラムフォーマット

3.2 データベース設計

本節ではデータベースの設計について述べる．本学のネットワークは，イーサネットにより構成されているため，データベーステーブルは以下 8 つを作成した．

a) header テーブル
表 1 の 1，2，3 のデータを格納する．2.4 節で述べたように，sFlow データグラムにはルータにより観測されたパケットデータ以外に，計測機器自身の情報が含まれるデータグラムもある．それらすべての sFlow データグラムに共通してこの 3 項目のヘッダが付加されている．今後の拡張を考慮して，別テーブルとした．

b) flow sample テーブル
表 1 の 5 から 11 までの項目を格納している．

c) ethernet ヘッダテーブル

d) IPv4 ヘッダテーブル

e) IPv6 ヘッダテーブル

f) TCP ヘッダテーブル

g) UDP ヘッダテーブル

h) payload テーブル

表 1 各項目の説明

No	説　明
1	sFlow データグラムのバージョン
2	データグラムの通し番号
3	sFlow データグラムが作成されたインタフェースが起動してからの経過時間 （現在時刻ではない）
4	sFlow データグラムの種類
5	計測機器に複数のインタフェースがある場合の， 各々のインタフェースが作成した sFlow データグラムの通し番号
6	sFlow データグラムが作成されたインタフェース情報
7	サンプリングレート
8	取得対象となったパケット数
9	取りこぼしたパケット数
10	取得パケットが受信されたインタフェース番号
11	取得パケットが送信されたインタフェース番号
12	sFlow データグラム中に含まれる観測パケットの数
以下，取得されたパケット上位 256byte までの RAW データ （イーサネットフレーム，FDDI フレームなど）	

c から h のテーブルへは sFlow データグラムに含まれるパケットの各ヘッダおよびペイロード部を保存する．それらのデータは，データベースへ蓄積するソフトウェアで独自にデコードされる．

3.3 計測環境

本学では，学外との境界に HITACHI の GS4000 を導入している．このルータは sFlow エージェントとして動作する．学内から学外への通信はこのルータを経由することになり，学内からの通信，学外からの通信を観測することができる．また，学内にも数台，sFlow に対応した機器を設置しており，各々sFlow エージェントとして設定されている．

sFlow コレクタには一般的なコンピュータを利用している．構成を表 2 に示す．

表 2 コレクタのスペック

CPU	Pentium III 1.2GHz
Memory	512MB
Hard Disk	80GB
NIC	100Mbps
OS	Slackware Linux 10.1
RDBMS	MySQL 4.0.23a

4. NetFlow を利用したネットワーク監視システムへの適応

本章では，本学で 2003 年 5 月より行っている NetFlow を利用したネットワークトラフィック監視システム[8]に sFlow を利用したネットワークトラフィック監視システムの適応を述べる．

sFlow を利用したネットワーク計測によって，データを 1 ヶ月間貯めた結果，レコード数はおよそ 240 万レコードであった．データ容量はおよそ 700MB であった．その 1/4 程度は payload 部のデータである．1 年間後のデータ容量の概算は 10GB 程度であることから，sFlow を利用したネットワーク計測でも長期間の計測が可能であるといえる．

図 2 は sFlow を利用したネットワーク監視システムで得られたデータを，NetFlow を利用したネットワーク監視システ

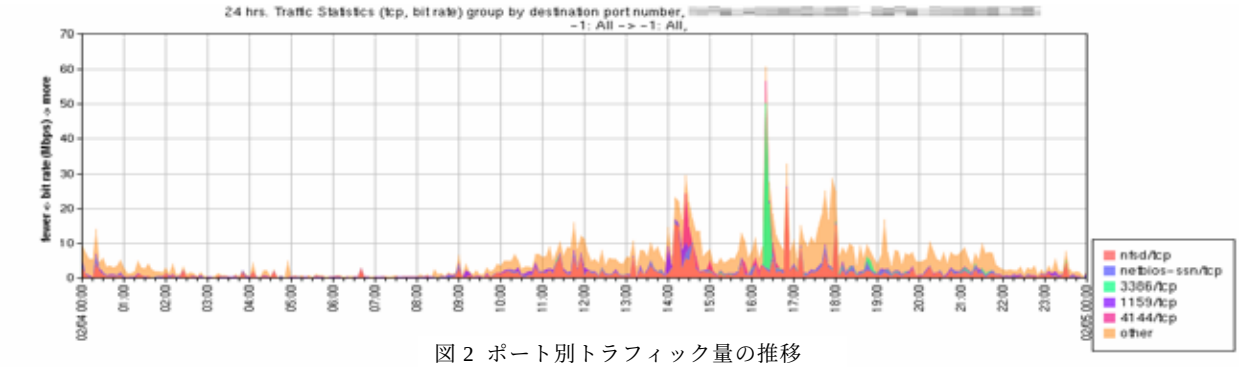

図 2 ポート別トラフィック量の推移

ムで利用しているトラフィック可視化ツールに適応させた例である．NetFlow を利用した計測により蓄積しているデータベースのテーブル構成とは異なるものの，取得できるデータは，Layer 2 から Layer 4 のヘッダと同等のものである．そのため SQL クエリを若干変更させるのみで容易に適応させることができる．

図 3 は，2005 年 5 月 2 日に TCP 6667 ポートを利用して通信しているホストで，通信パケット数の多い順に抽出した例である．6667 ポートは通常 IRC で利用するポートである．しかし bot[3]の操作に利用される事も多い．通信数が極端に多いホストは bot の感染疑いがある．図 4 は bot が活動している疑惑のあるホストを検出した例である．図中の pkt_count は実際観測されたパケット数であり，estimation はサンプリングレートから求めた実際の通信パケット数の予測値である．

実際，図 3 で通信パケット数 1 位に抽出されたホストは，w32/sdbot.worm.gen という bot に感染していた．しかしその他のホストに関しては false positive であった．

	src addr	dstport	pkt_count	estimation
	163.221.	6667	48	393216
	163.221.	6667	12	98304
	163.221.	6667	9	73728
	163.221.	6667	5	40960
	163.221.	6667	5	40960
	163.221.	6667	5	40960
	163.221.	6667	3	24576
	163.221.	6667	2	16384
	163.221.	6667	2	16384
	163.221.	6667	1	8192
	163.221.	6667	1	8192
	163.221.	6667	1	8192

図 3 bot 感染疑いホストの検出

	src addr	dstport	pkt_count	estimation
	163.221.	1433	388	3178496

図 4 bot 活動疑惑ホストの検出

5. NetFlow を利用したネットワーク監視システムの問題点とその解決

前章で述べたように， NetFlow を利用したネットワーク監視システムへの適応は可能であった．しかし，このシステムの問題点は，ヘッダ情報のみを用いて異常検出を行うため，本来異常ではない通信も異常と判断してしまう false positive の率が高いことにある．そこで，前章で述べたような異常の疑いのある通信データに対して IDS やウイルススキャンプログラムを適応することができれば，false positive 率の低下が期待できる．

本章においては，sFlow を利用したネットワーク監視システムで蓄積されるデータを IDS の一つである snort[6]へ適応させるための基礎実験と結果について述べる．

5.1 実験環境

本実験の実験ネットワークは図 5 に示すように，仮想内部ネットワークを定義し，その内部のノードに対して異常パケットを送信する．

エージェントには，Foundry Networks 社の FastIron Edge Switch2402 を用い，仮想ネットワーク側のイーサネットインタフェースで sFlow を計測するように設定した．

次に，異常パケット生成マシンのスペックを表 3 に示す．コレクタは表 2 に示したスペックのものと同じものを利用する．

異常パケットの生成には，hping2[4]を利用する．このツールは様々なパケットを生成することができ，firewall の設定のテストにも利用される．

IDS には，snort version 2.20 を用いる．snort は，フリーで利用することができ，事前に用意されているルールパターンに該当するトラフィックをリアルタイムに検出することができる．

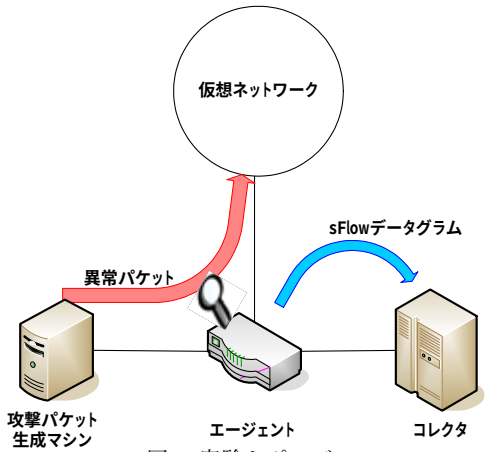

図 5 実験トポロジ

今回の実験では，snort のルールパターンは 2004 年 11 月現在の最新版を利用した．それらのルールパターンに加え，特定の文字がパケットに含まれた場合にアラートを発生させるルールを，実験用に独自で追加した．図 6 に追加したルールを示す．

表 3 異常パケット生成マシンスペック

CPU	Pentium4 2.53GHz
Memory	1024MB
Hard Disk	80GB
NIC	100Mbps
OS	Linux
ソフトウェア	hping2

```
alert tcp $EXTERNAL_NET any <> $HOME_NET 81 (msg:"appeared THE ALERT!!"; flow:stateless; content: "alert_alert"; classtype:misc-activity; sid:9999; rev:4;)
```

図 6 独自に追加したルール

5.2 実験方法

hping2 を利用し, 仮想ネットワーク内のクライアントに対して以下の 4 つのパターンのパケットを送信する.

1. TCP 宛先 port 番号 0 に対するパケット
2. 1 と同様のパケットで，パケット長が 256byte より大きいパケット
3. ペイロード部分に特定の文字列を含めた，パケット長 256byte 以下であるパケット
4. 3 と同様のパケットで, パケット長が 256byte より大きいパケット

sFlow により記録されるパケットはその仕様上，256byte に切りつめられる．そのため，本実験ではパケット長が 256byte 以下の場合と 256byte より大きい場合に分けて実験を行った.

以上 4 つの場合に分け，それぞれのパケットを一定時間送信し，エージェントに sFlow データグラムを送信させる．コレクタのデータベースに十数レコードが蓄積されたところで送信を停止する．データベースにアラートの対象となるパケット情報がいくつ蓄積されたかを確認した後，データベースの情報からパケットを復元し snort に適応させる.

これらのパケットを直接 snort に適応させた場合，パターン 1，2 のパケットは，"BAD-TRAFFIC tcp port 0 traffic" として検出される．パターン 3，4 は独自に追加したルールにより検出される.

5.3 実験結果

表 4 実験結果

実験パターン	アラート対象レコード数	実アラート数
パターン 1	13	13
パターン 2	11	11
パターン 3	12	12
パターン 4	12	**0**

表 4 に実験結果を示す. パターン 1 から 3 のパケットでは，データベースに蓄積された異常パケットに該当する該当レコード数と snort に適応させた場合のアラート数は一致した．しかしパターン 4 のパケットではアラートは発生しなかった.

以上の結果から，ヘッダのみを捜査するルールでの検出では，sFlow によりパケットが途中で切られていたとしても検出に問題がないことがわかった．しかし，ペイロードを捜査するルールでは，そのまま適応したとき，パケットサイズが 256byte より大きい場合，検出することができなかった．理由としては，パターン 4 のパケットデータをデータベースより復元すると，そのパケットの IP ヘッダに記録されているデータ長と，実際 snort に適応されたデータ長が異なる．snort はこのパケットをフラグメントされたパケットと判断してしまうようである．このため snort は，本来，後に続くべきパケットの到着を待つため，判断を見送る．しかし，実際はフラグメントが行われていないので，待ってもパケットが到達することはない．そのため，最終的に判断できずにアラートが発生しないと考えられる.

そこで，再度ヘッダ情報の修正を行ってパターン 4 の情報を復元し snort に適応させた. IP ヘッダのデータ長を実際のデータベースに記録されているデータ長（TCP ヘッダ長＋ペイロード長）に変更し復元した．この場合では，データベースに記録されたアラート対象のレコード数である 12 のアラートが発生した.

本章では, 蓄積されたデータを既存 IDS に適応させられることを示すことができた．また，データに若干修正を加えるのみで，正常にアラートが発生しなかったレコードに対してもアラートを発生させることができることを示せた.

6. おわりに

本稿では, 我々が開発した NetFlow を利用したネットワーク監視システムに対し sFlow を利用したネットワーク監視システムが適応可能であることを示した．また，NetFlow を利用したネットワーク監視システムの問題点を補うために，sFlow で取得可能なパケットデータを利用した IDS による異常トラフィック検知を行うための基礎実験を行った．これを既存の解析手法と組み合わせることで, 問題点であった false positive 率の減少が期待できる.

しかし，本稿においては，単一パケットを捜査するだけでアラートを発生させることができる単純なルールを適応させたのみで，セッション情報を利用するルールでは評価していない．今後 bot などの新しい脅威への対応が必要である.

課題として，

- セッション情報を利用するルールへの適応手法の確立
- 蓄積されたデータの復元手法毎，計測サンプリングレート毎による検出可能性，精度の調査，データ操作による副作用の検証
- 他の検出手法に適応可能な，汎用性のあるデータ復元手法の構築

などが挙げられる.

7. 参考文献

[1] MRTG，http://people.ee.ethz.ch/~oetiker/webtools/mrtg/

[2] NetFlow，http://www.cisco.com/warp/public/732/Tech/nmp/netflow/index.shtml

[3] bot，http://www.sans.org/rr/whitepapers/malicious/1299.php

[4] hping2，http://www.hping.org/

[5] sFlow.org，http://www.sflow.org/

[6] snort，http://www.snort.org/

[7] tcpdump，http://www.tcpdump.org/

[8] 藤井聖，中尾嘉宏，中村豊，藤川和利，砂原秀樹，“フローを用いた特定トラフィック検出システムの運用”, 情報処理学会研究報告，vol.2003，No.118，pp.103--108，November，2003