高速パケットキャプチャによるトラフィックの統計分析

2010SE005 安藤尚紀
指導教員：後藤邦夫

1 はじめに

私たちはインターネットの普及に伴い, webを利用した様々なサービスのおかげで生活が安定している. しかし, 近年, ネットワークは高速になっているため, 高速ネットワークでの瞬時に発生したすべてのパケットをキャプチャ出来ないという問題がある.

そこで本研究では, 高速でパケットキャプチャするLinuxのPF_RING[2] socketを用いて, 5分間通信トラフィックをパケット単位でキャプチャし, キャプチャしたデータを利用したベイズ推定 [4] により異常なトラフィックか否かを識別する手法を提案する. また, 実験によりその有効性を検証する. 実験ネットワークにはネットワークエミュレータソフトウェア「 CORE[1]」を使用する.

2 PF_PACKET socket, PF_RING socketの概要

本節では, PF_PACKET socketと本研究に使用するPF_RING socketの概要を述べる.

PF_PACKET socketとはパケットキャプチャする際に使用する標準的なソケットのことである. Linuxカーネルで TCP/IP スタックを経由してパケットキャプチャする. パケットを一定の速度でキャプチャするので, キャプチャ速度を越えてしまう速度での発生パケットを取りこぼしてしまう.

PF_RING socketとは高速パケットキャプチャするソケットのことである. 標準Linux kernelにはなく, kernel module追加し Linux kernel2.6.32以降で使用可能である. Linuxカーネルで TCP/IP スタックを経由せずにパケットキャプチャし, 受信したパケットをリングバッファ[1]にコピーする. PF_PACKET socketとは違い, パケットの取りこぼしが非常に少ない構造である.

PF_RING socketと PF_PACKET socketのキャプチャ性能の比較実験をし, 検証した. COREでホスト 1, ホスト 2, ホスト 3をルータに接続させる. ホスト 1とホスト 2でホスト 3に向けて 100,000 個の Flood pingを同時に実行し, ルータでパケットキャプチャを実行し, 10 回測定した. なお, パケットのデータ部サイズを最大に設定した. IP ヘッダは通常 20bytesで ICMP ヘッダは 8bytes, イーサネットの MTU が 1,500bytesなので最大データ部サイズは 1,500-(20+8)=1,472bytesになる. PF_RING socketと PF_PACKET socketの取りこぼしたパケット数の平均はそれぞれ 0bytes, 62.8bytesという結果になった. 使用した PC のスペックを表 1 に記載する. このような結果から, PF_PACKET socketより PF_RING socketの方がキャプチャ性能が優れているといえる.

[1]一定の領域の FIFO 型メモリを確保し, 先頭と末尾がつながった環状構造の配列のこと

表 1 実験に使用した PC のスペック

メーカー	富士通
型番	FWV-S8390
CPU	Core 2 Duo P8700(2.53GHz)
メモリ容量	2.0GByte
OS	Linux Ubuntu 12.04LTS 32bit
kernel version	3.2.0-52-generic-pae

3 ベイズ推定による提案手法

この節では, ベイズ推定による提案手法について述べる.

3.1 ベイズ推定の概要

ベイズ推定とはベイズの定理を使用して, ある事象が観測されたとき, ある仮説の事象を確率的な意味で決定することである. 事象の確率変数 A, B に対して, ベイズの定理

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

を用いる. A は仮説の確率変数, B は状況の確率変数である. ある状況を観測したとき, 仮説 A が成り立つ確率は $P(A|B)$ である. 推定手順は, $A = a_i(i = 0, 1, 2, \cdots, n)$ として, P_i を求める. ある状況 $B = b_j(j = 0, 1, 2, \cdots, n)$ を観測したとき,

$$P_i = P(b_j|a_i)P(a_i) \tag{2}$$

となり, 観測値で計算した P_i を最大とする i を推定事象とする.

3.2 ベイズ推定による異常トラフィック判別方法

本研究では [3] に習い, 次のような確率変数を考える.

- C: プロトコルと宛先 IP アドレスと宛先ポート番号の組み合わせ, あるいはプロトコルが ICMP の場合はプロトコルと宛先 IP アドレスと Type と Code の組み合わせのクラス.
- E_c: トラフィックの状態のクラス. E_c の取り得る値は, 正常なトラフィック $E_c = e_0$ と, 異常なトラフィック (発信元 IP アドレスが 1 種類)$E_c = e_1$ と, 異常なトラフィック (発信元 IP アドレスが多種類)$E_c = e_2$ の 3 種類とする.
- T_c: あるクラスの直前のパケットとの到着間隔時間の平均のクラス. T_c の取り得る値は, 1msec 以上 t_0, 100 μ sec 以上 1msec 未満 t_1, 100 μ sec 未満 t_2 の 3 種類とする.
- S_c: あるクラスの発信元 IP アドレスの種類の数のクラス. S_c の取り得る値は, 1 種類 s_0 と, 2 種類から 9 種類 s_1 と, 10 種類以上 s_2 の 3 種類とする.

これらを用いて, 事象の確率変数 A, B に対して (1) より ,

$$P(E_c|T_c, S_c) = \frac{P(T_c, S_c|E_c)P(E_c)}{P(T_c, S_c)} \qquad (3)$$

として, E_c がどんな値をとるか推定する. 事前に $P(T_c, S_c|E_c)$ および $P(E_c)$ を与える. $P(T_c, S_c|E_c)$ に与えた値を表 2 に記載する. なお, 表 2 に記載されていない確率は 0 とする. $P(e_0)$, $P(e_1)$, $P(e_2)$ の値をそれぞれ $\frac{8}{10}$, $\frac{1}{10}$, $\frac{1}{10}$ と与える. 確率の値は, 一般のブラウジングを考慮して決定した.

表 2 $P(t_i, s_i|e_i)$ の確率表

$P(t_0, s_2\|e_0)$	$\frac{76}{100}$
$P(t_1, s_0\|e_0)$	$\frac{26}{100}$
$P(t_2, s_2\|e_0)$	$\frac{8}{100}$
$P(t_1, s_0\|e_1)$	$\frac{35}{100}$
$P(t_2, s_0\|e_1)$	$\frac{65}{100}$
$P(t_2, s_1\|e_2)$	$\frac{24}{100}$
$P(t_2, s_2\|e_2)$	$\frac{76}{100}$

4 本研究で作成したプログラムの概要

本節では, 本研究に使用する 2 種類のプログラムの概要を述べる.

異常なトラフィックを生成するために, テスト用トラフィック発生プログラムを作成した. 任意の IP アドレスからある IP アドレスにパケット送信間隔時間 (msec) を指定し, 特定のパケットを送信するプログラムである. プロトコルは IPv4 のネットワークの TCP で, ポート番号は 80, 433 のみとする. これは, トラフィックのほとんどはポート番号 80(HTTP) やポート番号 443(HTTPS) を通過するが, これらのポートをすべてのトラフィックに出入り可能にすると不正なアプリケーションなどがネットワークに出入りすることがあり , この場合暗号化が死角となるからである.

パケットを分類するために, パケット分類プログラムを作成した. PF_ RING socket を用いてパケットキャプチャを実行し , C, T_c および S_c を出力するプログラムである.

5 実験結果

本節では, 異常トラフィックの検出実験について述べる. CORE で, 図 1 のようなネットワークを構築する. attacker から target にテスト用トラフィック発生プログラムを実行し , 同時に router でパケット分類プログラムを実行した. パケット分類プログラムの実行結果を次に示す.

パケット分類プログラムの実行結果

TCP|10.0.1.20|80 T_c: 0.051097msec S_c: 25

T_c が 0.051097msec なので t_2 と決定され, S_c が 25 なので s_2 と決定される. 第 3.2 節より ,

$P_0 = P(T_c = t_2, S_c = s_2|E = e_0)P(E = e_0) = \frac{8}{100} \times \frac{8}{10}$

$P_1 = P(T_c = t_2, S_c = s_2|E = e_1)P(E = e_1) = \frac{0}{100} \times \frac{1}{10}$

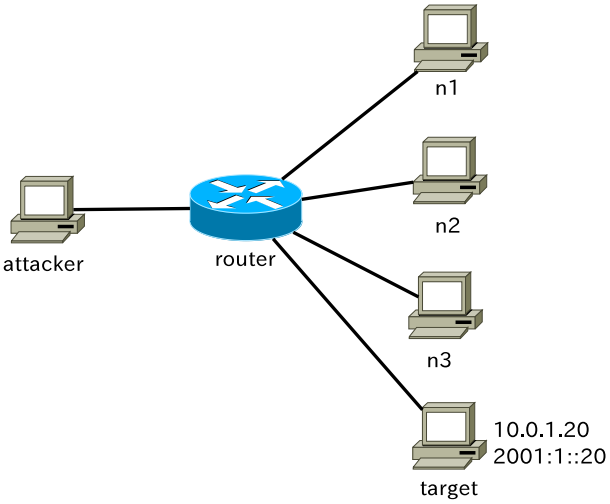

図 1 本研究の実験のシステム概要

$P_2 = P(T_c = t_2, S_c = s_2|E = e_2)P(E = e_2) = \frac{76}{100} \times \frac{1}{10}$

$P_1 = 0 < P_0 = \frac{64}{1000} < P_2 = \frac{76}{1000}$

したがって, 異常なトラフィック $E_c = e_2$(発信元 IP アドレスが多種類) と推定された.

6 おわりに

本研究において, 通信トラフィックをパケット単位での状態を識別し , トラフィックの状態を判別する手法を提案した. パケット分類プログラムを実行し , 出力データをベイズ推定することにより , トラフィックの状態判別の良好な判別結果を得た. この結果は, サーバの管理者にとって, ネットワークの状態を把握できる点で役に立つのではないかと考えられる. また, PF_RING socket は, 高速ネットワークでのパケットの取りこぼしを非常に減少させることが分かった. 今後の課題を箇条書きで述べる.

- ブラウジングやファイルダウンロードなど, 特定された動作を考慮する.
- 確率変数のクラス分けをより細かくする.
- 実際のネットワークでの実験.

上記の課題を介した実験により , 異常なトラフィックを発見する技術が進歩すると考えられる.

参考文献

[1] Ahrenholz, J.: Comparison of CORE Network Emulation Platforms, *Proc. of IEEE MILCOM Conference*, IEEE, pp. 864–869 (2010).

[2] Deri, L.: Improving Passive Packet Capture: Beyond Device Polling, *Proc. of SANE* (2004).

[3] 但馬康宏, 小谷善行, 寺田松昭: ベイズ推定によるインターネット通信の特徴抽出と適応的制御, 電子情報通信学会第 16 回データ工学ワークショップ第 3 回日本データベース学会年次大会 DEWS2005 論文集 (2005).

[4] 樋口知之: 予測にいかす統計モデリングの基本 ベイズ統計入門から応用まで, 講談社 (2011).