A Signal Analysis of Network Traffic Anomalies

Paul Barford, Jeffery Kline, David Plonka and Amos Ron

Abstract—**Identifying anomalies rapidly and accurately is critical to the efficient operation of large computer networks. Accurately characterizing important classes of anomalies greatly facilitates their identification; however, the subtleties and complexities of anomalous traffic can easily confound this process. In this paper we report results of signal analysis of four classes of network traffic anomalies: outages, flash crowds, attacks and measurement failures. Data for this study consists of IP flow and SNMP measurements collected over a six month period at the border router of a large university. Our results show that wavelet filters are quite effective at exposing the details of both ambient and anomalous traffic. Specifically, we show that a pseudo-spline filter tuned at specific aggregation levels will expose distinct characteristics of each class of anomaly. We show that an effective way of exposing anomalies is via the detection of a sharp increase in the local variance of the filtered data. We evaluate traffic anomaly signals at different points within a network based on topological distance from the anomaly source or destination. We show that anomalies can be exposed effectively even when aggregated with a large amount of additional traffic. We also compare the difference between the same traffic anomaly signals as seen in SNMP and IP flow data, and show that the more coarse-grained SNMP data can also be used to expose anomalies effectively.**

I. Introduction

Traffic anomalies such as failures and attacks are commonplace in today's computer networks. Identifying, diagnosing and treating anomalies in a timely fashion is a fundamental part of day to day network operations. Without this kind of capability, networks are not able operate efficiently or reliably. Accurate identification and diagnosis of anomalies first depends on robust and timely data, and second on established methods for isolating anomalous signals within that data.

Network operators principally use data from two sources to isolate and identify traffic anomalies. The first is data available from Simple Network Management Protocol (SNMP) queries to network nodes. This Management Information Base (MIB) data is quite broad, and mainly consists of counts of activity (such as number of packets transmitted) on a node. The second type of data available is from IP flow monitors. This data includes protocol level information about specific end-to-end packet flows which make it more specific than SNMP data. The combination of these types of data provides a reasonably solid measurement foundation for anomaly identification.

Unfortunately, current best practices for identifying and diagnosing traffic anomalies are almost all ad hoc. These consist mainly of visualizing traffic from different perspectives and identifying anomalies from prior experience. While a variety of tools have been developed to automatically generate alerts to failures, it has generally been difficult to automate the anomaly identification process. An important step in improving the capability of identifying anomalies is to isolate and characterize their important features.

A road map for characterizing broad aspects of network traffic was outlined in [1]. In this paper, we restrict our focus to one aspect of that work and report results of a detailed *signal analysis* of network traffic anomalies. Our analysis considers the time-frequency characteristics of IP flow and SNMP data collected at the border router of the University of Wisconsin-Madison over a 6 month period. Included with these data is a catalog of 109 distinct traffic anomalies identified by the campus network engineering group during the data collection period. This combination of data enabled us to focus our efforts on how to employ filtering techniques to most effectively expose local frequency details of anomalies.

To facilitate this work, we developed the Integrated Measurement Analysis Platform for Internet Traffic (IMAPIT). IMAPIT contains a data management system which supports and integrates IP flow, SNMP and anomaly identification data. IMAPIT also includes a robust signal analysis utility which enables the network traffic data to be decomposed into its frequency components using a number of wavelet and framelet systems. More details of IMAPIT are given in Sections IV and V.

Initially, we analyzed a variety of traffic signals by applying general wavelet filters to the data. Wavelets provide a powerful means for isolating characteristics of signals via a combined time-frequency representation[1]. We tested the wavelet analysis by applying many different wavelet systems to traffic signals to determine how to best expose the characteristics of anomalies recorded therein. We accepted the constraint that our flow and SNMP data was collected at five minute intervals, thereby precluding analysis on finer timescales. Nevertheless, we were able to select a wavelet system and develop algorithms that effectively expose the underlying features of both ambient and anomalous traffic.

Not surprisingly, our analysis shows clear daily and weekly traffic cycles. It is important to be able to expose these components so that anomalous traffic can be effectively isolated. Our analysis then focused on anomalies by separating them into two groups based on their observed duration. The first group consisted of flash crowd events which were the only long-lived events in our data set - these typically span up to a week. Flash crowd anomalies are effectively exposed using the low frequency representation in our system. The second group of anomalies are those that were short-lived and consisted of network failures, attacks, and other events. These short-lived

P. Barford and A. Ron are members of the Computer Sciences Department at the University of Wisconsin, Madison. E-mail: pb,amos@cs.wisc.edu. J. Kline is a member of the Mathematics Department at the University of Wisconsin, Madison. E-mail: kline@math.wisc.edu. D. Plonka is a member of the Division of Information Technology at University of Wisconsin, Madison. E-mail: plonka@doit.wisc.edu

[1] Standard Fourier analysis only enables localization by frequency.

anomalies are more difficult to expose in data due to their similarity to normal bursty network behavior. We found that these signals could be effectively exposed by combining data from the mid and high frequency levels. Our investigation of which combinations of data best expose anomalies included comparing SNMP to IP flow data, breaking down flow data by packet, byte and flow metrics, and measuring variations in the packets' average size.

One important test that we developed for exposing short-lived events was based in computing the normalized local variance of the mid and high frequency components of the signal. The intuition for this approach is that the "local deviation" in the high frequency representation exposes the beginning and end of short-lived events and the local variability in the mid frequency filters expose their duration. Large values of these local variances indicates a sharp *unpredictable* change in the volume of the measured quantity. Our *Deviation Scoring* method described in Section IV is a first step at attempting to automate anomaly detection (which is now typically done by visual inspection of time-series network traffic plots) through the use of multi-resolution techniques. Employing this method on our data over a number of weeks actually exposed a number of true anomalies (verified post-mortem by network engineers) that had not been cataloged previously.

While the majority of our work focused on identifying anomalies in aggregate traffic at the campus border router, the source and destination address in the IP flow data allows us to isolate anomalies at different points in the network (by pruning away traffic from various subnets). As you move closer to the source of an anomaly, the event typically becomes more pronounced in the data and thus easier to expose. However, if the event takes place at a point in the network where there is lower aggregation of traffic then there is typically more variability in the ambient traffic and, as a result, the task of isolating the anomaly signal becomes more difficult. We show that our methods work well whether the measurement point is close to or distant from the point of the anomaly.

This paper is organized as follows. In Section III we describe the data sets we use in this work. We also describe current best practices employed by network operators for general anomaly detection. In Section IV we describe our signal analysis methods and the IMAPIT framework. In Section V we present the results of our analysis and discuss their implications. We evaluate the performance of our anomaly detection method in Section VI, and then summarize, conclude and discuss future work in Section VII.

II. Related Work

General properties of network packet traffic have been studies intensely for many years - standard references include [2], [3], [4], [5]. Many different analysis techniques have been employed in these and other studies including wavelets in [6]. The majority of these traffic analysis studies have been focused on the typical, packet level and end-to-end behavior (a notable exception being [7]). Our focus is mainly at the *flow level* and on identifying frequency characteristics of anomalous network traffic.

There have been many prior studies of network fault detection methods. Example include [8], [9], [10]. Feather *et al.* use statistical deviations from normal traffic behavior to identify faults [11] while a method of identifying faults by applying thresholds in time series models of network traffic is developed in [12]. These studies focus on accurate detection of deviations from normal behavior. Our work is focused on identifying anomalies by removing first from the signal its predictable, ambient part, and only then employing statistical methods. Wavelet are used for the former task.

Detection of black-hat activity including denial-of-service (DoS) attacks and port scan attacks has also been treated widely. Methods for detecting intrusions include clustering [13], neural networks [14] and Markov models [15]. Moore *et al.* show that flow data can be effective for identifying DoS attacks [16]. A number of intrusion detection tools have been developed in recent years in response to the rise in black-hat activity. An example is Bro [17] which provides an extensible environment for identifying intrusion and attack activity. Our work complements this work by providing another means for identifying a variety of anomalous behaviors including attacks.

We identify flash crowds as an important anomaly category. The events of September 11, 2001 and the inability of most online news services to deal with the offered demand is the most extreme example of this kind of behavior. While infrastructure such as content delivery networks (CDNs) have been developed to mitigate the impact of flash crowds, almost no studies of their characteristics exist. A recent study on flash crowds is by Jung *et al.* in [18]. That work considers flash crowds (and DoS attacks) from the perspective of Web servers logs whereas ours is focused on network traffic. Finally, *cooperative pushback* is proposed in [19] as a means for detection and control of events such as flash crowds.

III. Data

A. The Measurement Data

Our analysis is based on two types of network traffic data types: SNMP data and IP flow data. The source of both was a Juniper M10 router which handled all traffic that crossed the University of Wisconsin-Madison campus network's border as it was exchanged with the outside world. The campus network consists primarily of four IPv4 class B networks or roughly 256,000 IP addresses of which fewer than half are utilized. The campus has IP connectivity to the commodity Internet and to research networks via about 15 discrete wide-area transit and peering links all of which terminate into the aforementioned router.

The SNMP data was gathered by MRTG [20] at a five minute sampling interval which is commonly used by network operators. The SNMP data consists of the High Capacity interface statistics defined by RFC2863 [21] which were polled using SNMP version 2c. This analysis used the byte and packet counters for each direction of each wide-area link, specifically these 64-bit counters: ifHCInOctets, ifHCOutOctets, ifHCInUcastPkts, and ifHCOutUcastPkts.

The flow data was gathered using flow-tools [22] and was post-processed using FlowScan [23]. The Juniper M10 router was running JUNOS 5.0R1.4, and later JUNOS 5.2R1.4, and was configured to perform "cflowd" flow export with a packet sampling rate of 96. This caused 1 of 96 forwarded packets to be sampled, and subsequently assembled into flow records similar to those defined by Cisco's NetFlow [24] version 5 with similar packet-sampling-interval and 1 minute flow active-timeout. The packet and byte counts computed from those flow records were then multiplied by the sampling rate to approximate the actual byte and packet rates. We have not attempted to formally determine the accuracy of packet-sampling-based flow measurements as compared with the SNMP measurements. However, it is common to use such measurements in network operations.

Both the SNMP and flow data were post-processed to produce rate values and stored using the RRDTOOL [20] time-series database. The archives were configured to retain values at five minute granularity from September 25, 2001 through April 4, 2002. Internet service providers which bill customers based upon 95th percentile peak interface usage have a similar retention policy for SNMP data. However, most other network operators retain the five-minute granularity data for only about two days (50 hours for MRTG, by default), after which that data is coalesced into averages over a increasingly longer time intervals; typically 30 minute, 2 hour, and 24 hour averages. For the campus, with approximately 600 IP subnets, this set of data resulted in a database of approximately 4GB in size. The collected flow records were retained to validate the results of the analysis. They were collected at five minute intervals resulting in about 60,000 compressed files of approximately 100GB in total combined size.

Though uncommon, our analysis also considered the average IP packet size, as computed from corresponding byte and packet rates. Because many applications have typical packet sizes, often bimodal with respect to requests and responses or data and acknowledgments, analysis of this metric occasionally exposes application usage even when only SNMP-based byte and packet interface rate statistics are available.

In parallel with the collection of the measurement data, a journal of known anomalies and network events was maintained. The log entries in this journal noted the event's date and time, and a one-line characterization of the anomaly. Furthermore, a simple nomenclature was used to label events as one of these types:

- **Network**: A network failure event or temporary misconfiguration resulting in a problem or outage. For instance: router software spontaneously stopped advertising one of the campus class B networks to campus BGP peers.
- **Attack**: Typically a Denial-of-Service event, usually flood-based. For instance: an outbound flood of 40-byte TCP packets from a campus host that has had its security compromised and is being remotely controlled by a malicious party.
- **Flash**: A flash crowd [18] event. For instance: the increase in outbound traffic from a campus `ftp` mirror server following a release of RedHat Linux.
- **Measurement**: An anomaly that we determined not to be due to network infrastructure problems nor abusive network usage. For example: a campus host participating in TCP bulk data transfer with a host at another campus as part of a research project. Problems with the data collection infrastructure itself were also categorized as "Measurement" anomalies. These include loss of flow data due to router overload or unreliable UDP NetFlow transport to the collector.

TABLE I

TYPES AND COUNTS OF NETWORK ANOMALY EVENTS IN THE TRAFFIC DATABASE USED IN THIS STUDY.

Anomaly Type	Count
Network	41
Attack	46
Flash Crowd	4
Measurement	18
Total	109

In this way, a total of 168 events were identified and a subset researched and tagged by the engineers operating the campus network. Table I shows the distribution of types among the 109 tagged events. All flash crowd events occuring during the measurement period were selected along with a sampling of anomalies in the other three categories based on those that had the most detailed description in the operator's journal. While the journal did not record every traffic anomaly during the measurement period, it acted as a unique road map for exploring the raw traffic measurement data, and provided a basis for determining if the anomalies could detected or characterized automatically.

B. Best Current Practice

Experienced network operators often employ effective, but ad hoc, methods of problem determination and anomaly detection. These techniques rely heavily on an operator's experience and persistent personal attention.

Modern network management systems (NMS) software provides two common tools for handling SNMP data. The first, and ostensibly most-used, is a graphing tool capable of continuously collecting and plotting values from the MIBs. It is not uncommon for a network operators to fill their workstations' screens with plots of traffic as it passes through various network elements.

The second is an alarm tool, which periodically performs tests on collected values and notifies operators accordingly. Such tools are based on locally authored rules, perhaps augmented by heuristics provided by the NMS vendor. These rules are often rudimentary conditional threshold tests such as, "if (router.interface1.utilization $>$ 50%) then notify". The resulting expert knowledge expressed in these rules is not necessarily portable to any other network environment.

Tools for handling flow data are less mature. Freely-available tools such as those employed in our work, have achieved a certain level of popularity among operators of enterprise and large networks. These tools leverage existing SNMP experience by converting detailed flow-export records into familiar time-series data. Tabular data, compiled by either commercial and freely-available tools, is occasionally used as well.

The major deficiency of these tools is the amount of expert local knowledge and time required to setup and use them pervasively. For instance, we collected about 6,000 unique time-series metrics from just a single network element, namely our campus border router. This amount of data prohibits visual inspection of graphs containing plots of all but a small subset of those metrics.

IV. Methods

A. Wavelet Analysis

A typical input of our analysis platform is a string of Internet traffic measurements. One of the basic principles of our methodology is the treatment of the measurement string as a *generic signal*, ignoring, at least to a large degree, the semantics of signal (such as the content of the packet header), the instrumentation used (e.g. SNMP vs. FlowScan), the quantity which is being measured (packet count, byte count, incoming traffic, outgoing traffic), or the actual subnet which is targeted. We do pay careful attention to time aggregation of the measurement (one measurement for each five minutes) in order to capture daily and weekly patterns or inconsistencies. An approach like the above is important since we would like to build a platform which is *portable*, and that can be *automated*. Any analysis tool that depends heavily on the nature of a particular local subnet will almost surely not be portable to other locations in the due to the heterogeneity of Internet traffic.

The basic tool we employ is *wavelet analysis*. The wavelet tool organizes the data into *strata*, a hierarchy of component "signals", each of which maintains *time* as its independent variable. The lower strata contain very sparse filtered information that can be thought of as sophisticated aggregations of the original data. We refer to that part of the representation as the *low-frequency* representation. In our algorithm, we derive one such representation, *i.e.* a single dataset that extracts the general slow-varying trends of the original signal. In contrast, the very high strata in the hierarchy capture fine-grained details of the data, such as spontaneous variations. These are referred to as the *high-frequency* strata.

Let us review with a bit more detail the so-called "wavelet processing". This processing is actually made of two complementary steps: the first is *analysis/decomposition* and the other is its inverse, the *reconstruction/synthesis* process.

Analysis: The goal of the analysis process is to extract from the original signal the aforementioned hierarchy of derived signals. This is done as an iterative process. The input for each iteration is a signal x of length N. The output is a collection of two or more derived signals, each of which is of length $N/2$. Each output signal is obtained by convolving x with an specially designed *filter* F and then decimating every other coefficient of that convolution product. We denote by $F(x)$ the output signal so obtained. One of the special filters, denoted herein as L, has a smoothing/averaging effect, and its corresponding output $L(x)$ is the *low-frequency* output. The other filters, $H_1, \ldots, H_r$ ($r \geq 1$) are best thought of as "discrete differentiation", and a typical output $H_i(x)$ should capture only the "fine-grained details", *i.e.* of the high-frequency content of the signal x. The iterations proceed with the further decomposition of $L(x)$, creating the (shorter) signals $L^2(x), H_1L(X), \ldots, H_rL(x)$. Continuing in this manner, we obtain a family of output signals of the form $H_iL^{j-1}(x)$. The index j counts the number of low-pass filtering iterations applied to obtain the output signal: the larger the value of j, the lower the derived signal is in our hierarchy. Indeed, we refer to $H_iL^{j-1}(x)$ as belonging to the jth frequency level, and consider a higher value of j to corresponding to a lower frequency. If our original signal x consists of measurements taken at five minute intervals, then the derived signal $H_iL^{j-1}(x)$ consists of data values that are $2^j \times 5$ minutes apart one from the other. Thus, as j grows, the corresponding output signal becomes shorter and records a smoother part of the signal. The values of the derived signals $H_iL^{j-1}(x)$ ($i = 1, \ldots, r$, $j \geq 1$) are known as the *wavelet coefficients*.

For example, let us consider the case $j = 6$. At that level, the derived signal $H_iL^6(x)$ contains aggregated data values that are $2^6 \times 5 = 320$ minutes apart. At that aggregation level (if done correctly) we should not anticipate seeing subtle variations that evolve along, say, two or three hour duration; we will see, at best, a very blurry time-stamp of such variations. On the other hand, the coefficients at that level might capture well the variations between day and night traffic.

The *synthesis iterations* perform the inverse of the analysis: at each step the input signals for the iteration are $L^j(x), H_1L^{j-1}(x), \ldots, H_rL^{j-1}(x)$, and the output is the signal $L^{j-1}(x)$. This is exactly the inverse of the jth iteration of the analysis algorithm. By employing that step sufficiently many times, one recaptures the original signal.

One possible way of using the wavelet process is in "detection-only" mode. In this mode, one examines the various derived signals of the decomposition, and tries to infer from them information about the original signal.

Wavelet-based algorithms are usually more sophisticated and attempt to assemble a new signal from the various pieces in the decomposition. This is done by altering some of the values of some of the derived signals of the decomposition step and then applying reconstruction. The general idea is to suppress all the values that carry information that we would like to ignore. For example, if we wish only to view the fine-grained spontaneous changes in the data, we will apply a *threshold* to the entries in all the low-frequency levels, *i.e.* replace them by zeros.

The above description falls short of resulting in a well-defined algorithm. For example, suppose that we would like to suppress the day/night variation in the traffic. We mentioned before that such variations appear in frequency level 6 (and definitely in lower levels as well). But, perhaps that are also recorded in the derived signal at level 5? It turns out that there is no simple answer here. The wavelet tranform we describe here is one of many possible wavelet systems, each of which might provide a unique decomposition of the data. Unfortunately, choosing among the subtle details of each wavelet transform often requires an expert understanding of the performance of those wavelet decompositions. Their ultimate success depends on selecting a wavelet transform that suits the given application.

Time frequency-localization: approximation orders and vanishing moments. In a highly qualitative description, the selection of the wavelet transform should be based on a careful balance between it's *time localization* characteristics, and its *frequency localization* characteristics.

Time localization is a relatively simple notion that is primarily measured by the length of the filters that are employed in the transform. Long filters lead to excessive blurring in the time domain. For example, the use of long filters denies us the ability to easily distinguish between a very strong short-duration change in traffic volume, as opposed to a milder change of longer duration. Since, for anomaly detection, the ability to answer accurately the question "when?" is critical, we chose a wavelet system for very short filters.

Frequency localization. In our context, there are two characteristics of the wavelet system that may be regarded as belonging to this class.

One way to measure frequency localization is by measuring the number of *vanishing moments* that the analysis filters H_i possess. We say that the filter H_i has k vanishing moments if $\hat{H}(0) = \hat{H}'(0) = \ldots = \hat{H}^{(k-1)}(0) = 0$ where $\hat{H}$ is the Fourier series of H. In every wavelet system, every filter H_i has at least one vanishing moment. Filters with a low number (usually one or two) of vanishing moments may lead to the appearance of large wavelet coefficients at times when no significant event is occurring thus resulting in an increasing in the number of false positive alerts. In order to create a wavelet transform with high number of vanishing moments, one needs to select longer filters.

Another closely related way to measure frequency localization is via the *approximation order* of the system. We forgo explaining the details of this notion and mention only that the decision to measure frequency localization either via vanishing moments or by approximation order depends primarily on the objective and the nature of algorithm that is employed.

The last issue in this context is the *artifact freeness* of the transform. For many wavelet systems, the reconstructed (modified) signal shows "features" that have nothing to do with the original signal, and are artifacts of the of the filters used. Wavelet filters that are reasonably short and do not create such undesired artifacts are quite rare; thus, our need for good time localization together with our insistence on an artifact-free wavelet system narrowed the search for the "optimal" system in a substantial way.

The wavelet system we employ: We use a bi-frame version of a system known as PS(4,1)Type II (cf. [25]). This is a *framelet* system, *i.e.* a redundant wavelet system (which essentially means that r, the number of high-pass filters, is larger than 1; a simple count shows that, if $r > 1$, the total number of wavelet coefficients exceeds the length of the original signal). In our work, the redundancy itself is not considered a virtue. However, the redundancy provides us with added flexibility: it allows us to construct relatively short filters with very good frequency localization.

In our chosen system, there is one low-pass filter L and three high-pass filters H_1, H_2, H_3. The analysis filters are all 7-tap (i.e. each have 7 non-zero coefficients), while the synthesis filters are all 5-tap. The vanishing moments of the high-pass analysis filters are $2, 3, 4$, respectively, while the approximation order of the system is 4[2]. The "artifact freeness" of our system is guaranteed since our low-pass filters deviate only mildly from spline-filters (that perform pure multiple averages, and are the ideal artifact-free low-pass filters).

The analysis platform. We derive from a given signal x (that represents five-minute average measurements) three output signals, as follows. The description here fits a signal that has been measured for two months. Slightly different rules were employed for shorter duration signals (*e.g.* a signal measured for a week).

- The L(ow frequency)-part of the signal, obtained by synthesizing all the low-frequency wavelet coefficients from levels 9 and up. The L-part of the signal should capture patterns and anomalies of very long duration: several days and up. The signal here is very sparse (its number of data elements is approximately 0.4% of those in the original signal), and captures weekly patterns in the data quite well. For many different types of Internet data, the L-part of the signal reveals a very high degree of regularity and consistency in the traffic, hence can reliably capture anomalies of long duration (albeit it may blur various characteristics of the abnormal behavior of the traffic.)
- The M(id frequency)-part of the signal, obtained by synthesizing the wavelets coefficients from frequency levels $6, 7, 8$. The signal here has zero-mean, and is supposed to capture mainly the daily variations in the data. Its data elements number about 3% of those in the original signal.
- The H(igh frequency)-part of the signal is obtained by thresholding the wavelet coefficients in the first 5 frequency levels, *i.e.* setting to zero all coefficients whose absolute value falls below a chosen threshold (and setting to zero all the coefficients in level 6 and up). The need for thresholding stems from the fact that most of the data in the H-part consists of small short-term variations, variations that we think of as "noise" and do not aid us in our anomaly detection objective.

We close this section with two technical comments: while the theory of thresholding redundant representations is still in rudimentary form, it is evident to us that we should vary the thresholding level according to the number of vanishing moments in the filter (decreasing the threshold for the filter with high vanishing moments.) We have not yet implemented this technique. Finally, due to its high approximation order, our system cannot capture accurately sharp discontinuities in the data.

Detection of anomalies. While it is unlikely that a single method for detecting anomalies will be ever found[3], we have taken a first step at developing an automated method for identifying irregularities in the measured data. Our algorithm, which we call a *deviation score*, has the following ingredients:

1. Normalize the H- and M-parts to have variance one. Compute the local variability of the (normalized) H- and M-parts by computing the variance of the data falling within a moving window of specified size. The length of this moving window should depend on the duration of the anomalies that we wish to captured. If we denote the duration of the anomaly by t_0 and the time length of the window for the local deviation by t_1, we need, in the ideal situation, to have $q := t_0/t_1 \approx 1$. If the quotient q is too small, the anomaly may be blurred and lost. If the quotient is too large, we may be overwhelmed by "anomalies" that are of very little interest to the network operators. Our current experiment focuses on anomalies of duration 1-4 hours, and uses a moving 3-hour local deviation window. Shorter anomalies of

[2] Our initial assumption was that the Internet traffic is not smooth, and there might not be enough gain in using a system of approximation order 4 (had we switched to a system with approximation order 2, we could have used shorter filters). However, comparisons between the performance of the PS(4,1) Type II to the system RS4 (whose filters are all 5-tap, but whose approximation order is only 2), yielded a significant difference in performance.

[3] After all, there is not a single definition of "anomaly". Should we consider any change in the measured data to be an anomaly or only those that correspond to an identifiable change in network state?

sufficient intensity are also detected.
2. Combine the local variability of the H-part and M-part of the signal using a weighted sum. The result is the V(ariable)-part of the signal.
3. Apply thresholding to the V-signal. By measuring the peak height and peak width of the V-signal, one is able to begin to identify anomalies, their duration, and their relative intensity. We provide further details of our application of this technique in Section V.

While this approach to identifying anomalies that occur over periods of hours appears to be promising, it is only the first step in a process of automated anomaly detection based on the use of wavelet coefficients. Our choice of using scale dependent windowing to calculate deviation score is motivated by simplicity. This approach enabled us to easily quantify the significance of local events by using the reconstructed signal's local variability. In the future, we may find that direct use of combinations of wavelet and approximation coefficients (or other components) will be sufficient for accurate automated anomaly detection. To that end, as future work, we plan to investigate which components provide the best descrimination and to employ machine learning tools and techniques to develop more robust automated anomaly detectors. This approach will enable us to evaluate quantitatively which combinations of wavelet (or other) features provide the best detection capability.

B. *The IMAPIT Analysis Environment*

The IMAPIT environment we developed for this study has two significant components: a data archive and a signal analysis platform. The data archive uses RRDTOOL (mentioned in Section III) which provides a flexible database and front-end for our IP flow and SNMP data. The analysis platform is a framelet signal analysis and visualization system that enables a wide range of wavelet systems to be applied to signals.

Signal manipulation and data preparation in IMAPIT analysis was performed using a modified version of the freely-available LastWave software package [26]. In addition to wavelet decomposition, we implemented our deviation score method for exposing signal anomalies. Both flow and SNMP time-series data can be used as input to compute the deviation score of a signal.

Calculating the deviation score has four parameters: an M-window size, an H-window size, and the weights assigned to the M- and H-parts. We used only a single constant set of parameter values to produce the results in Section V. However, one can tune IMAPIT's sensitivity to instantaneous events by modifying the moving window size used in constructing the local deviation; a smaller window is more sensitive. The weights used on the M- and H-parts allow one to emphasize events of longer or shorter duration.

In our analysis, we found most anomalies in our journal had deviation scores of 2.0 or higher. We consider scores of 2.0 or higher as "high-confidence", and those with scores below 1.25 as "low-confidence". Where deviation scores are plotted in figures in Section V, we show the score as a grey *band* clipped between 1.25 and 2.0 on the vertical axis, as labeled on the right side. An evaluation of deviation scoring as a means for anomaly detection can be found in Section VI.

V. RESULTS

We decompose each signal under analysis into three distinct signals (low/mid/high). As a point of reference, if the signal under analysis is 1 week long (the period used to evaluate short-lived anomalies), the H-part is frequency levels 1,2,3; the M-part is frequency levels 4,5; the L-part is the remainder. If the signal is 8 weeks long (the period used to evaluate long-lived anomalies), the M-part is frequency levels 6,7,8; and the L-part is the remainder.

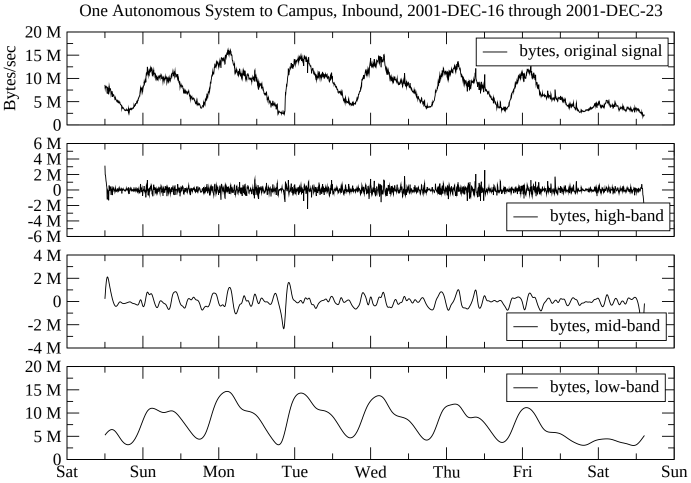

Fig. 1. Aggregate byte traffic from IP flow data for a typical week plus high/mid/low decomposition.

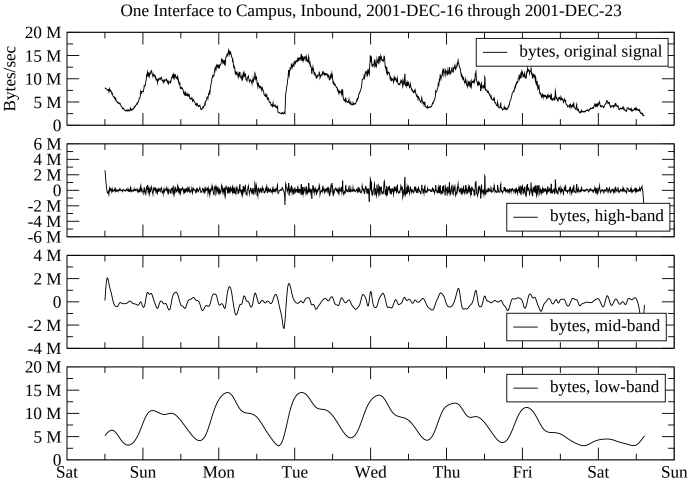

Fig. 2. Aggregate SNMP byte traffic for the same week as Figure 1 plus high/mid/low decomposition.

A. *Characteristics of Ambient Traffic*

It is essential to establish a baseline for traffic free of anomalies as a means for calibrating our results. Many studies describe the essential features of network traffic (*e.g.* [4]) including the standard daily and weekly cycles. Figure 1 shows the byte counts of inbound traffic to campus from the commodity

Internet Service Provider during a typical week. The figure also shows the wavelet decomposition of the signal into high, mid, and low-band components corresponding to the H-, M-, and L-parts discussed in Section IV. The regular daily component of the signal is very clear in the low band[4].

In Figure 2, we show the byte traffic for the same week at the same level of aggregation as measured by SNMP. In this case traffic was measured by utilizing high-capacity SNMP interface octet counters rather than by selecting the specific BGP Autonomous System number from the exported flow records. The decompositions in Figures 1 and 2 are nearly indistinguishable. The primary difference is slightly more high-frequency "jitter" in the flow-export-based signal[5].

B. Characteristics of Flash Crowds

The first step in our analysis of anomalies is to focus on flash crowd events. Our choice of investigating flash crowds first is due to their long lived features which should be exposed by the mid and low-band filters. This suggests that analysis of either SNMP or flow-based data is suitable, however we focus on flow-based data. Figure 3 shows the decomposition of eight weeks of outbound traffic from one of the campus' class-B networks which contains a popular ftp mirror server for Linux releases. During these weeks, two releases of popular Linux distributions occurred, resulting in heavy use of the campus mirror server. In this and subsequent figures, grey *boxes* were added by hand to focus the reader's attention on the particular anomaly (the position of each box was determined by simple visual inspection). Attention should again focus on the low-band signal. The low-band signal highlights each event clearly as well as the long-lived aspect of the second event.

Another way to consider the effects of flash crowds is from the perspective of their impact on the typical sizes of packets. The intuition here is that large data/software releases should result in an increase in average packet size for outbound HTTP traffic and therefore packet size may be an effective means for exposing flash crowds. Figure 4 shows eight weeks of outbound HTTP traffic and highlights another flash crowd anomaly from our data set. This anomaly was the result of network packet traces being made available on a campus web server. Curiously, for unrelated reasons, the server for this data set had its kernel customized to use a TCP Maximum Segment Size of 512. Both the mid-band and low-band signals in this figure show that the outbound HTTP packets from this server were, in fact, able to redefine the campus' average HTTP packet size. It is also interesting to note that the packet size signal becomes *more stable* (the signal has fewer artifacts) during this flash crowd event. This is particularly visible in the mid-band. Since flash crowds typically involve a single application, it seems likely that that application's packet size "profile" temporarily dominates.

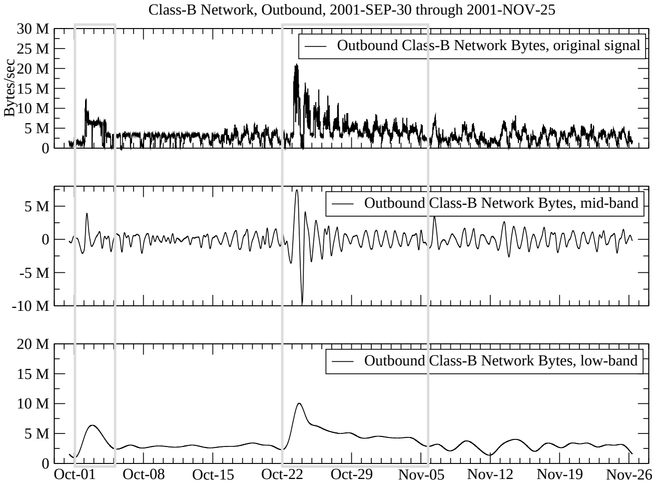

Fig. 3. Baseline signal of byte traffic for a one week on either side of a flash crowd anomaly caused by a software release plus high/mid/low decomposition.

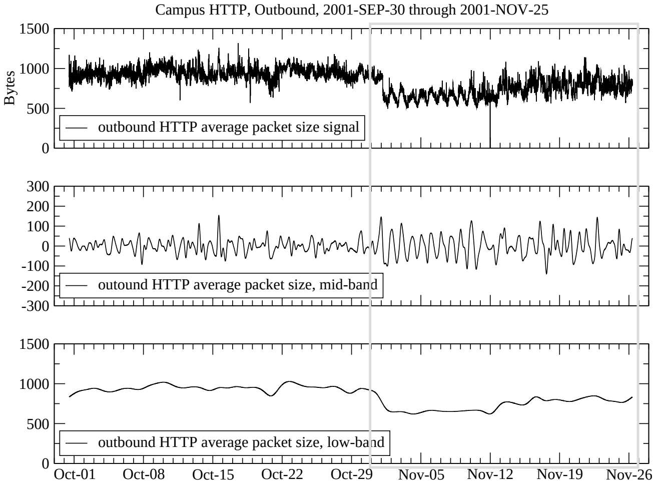

Fig. 4. Baseline signal of average HTTP packet sizes (bytes) for four weeks on either side of a flash crowd anomaly plus mid/low decomposition.

C. Characteristics of Short-term Anomalies

Short-term anomalies comprise attacks, network outages, and measurement anomalies. The coarse-grained (5 minute intervals) nature of our measurements complicates discrimination between these categories of anomalies, thus we consider them as a group. We evaluated 105 short-term anomalies using different combinations of data to determine how best expose their features (we present analysis of several examples of short-term anomalies to highlight their general features). In contrast to flash crowds, short-term anomaly features should be best exposed by mid-band and high-band filters which isolate short-timescale aspects of signals.

Figure 5 shows a decomposition of TCP flow counts which exposes two inbound denial-of-service (DoS) attacks that occurred during the same one week period. These two attacks were floods of 40-byte TCP SYN packets destined for the same campus host. Because the flood packets had dynamic source ad-

[4] We could easily expose the weekly component of the signal using higher aggregation filters, however weekly behavior is not important for the groups of anomalies we consider in this study.

[5] We will employ formal methods to quantify the difference between SNMP and flow signals and their decompositions in future work.

dresses and TCP port numbers, the flood was reported as many "degenerate" flows, having only one packet per flow. As predicted, the decomposition easily isolates the anomaly signal in the high and mid bands. By separating these signals from the longer time-scale behavior, we have new signals which may be amenable to detection by thresholding.

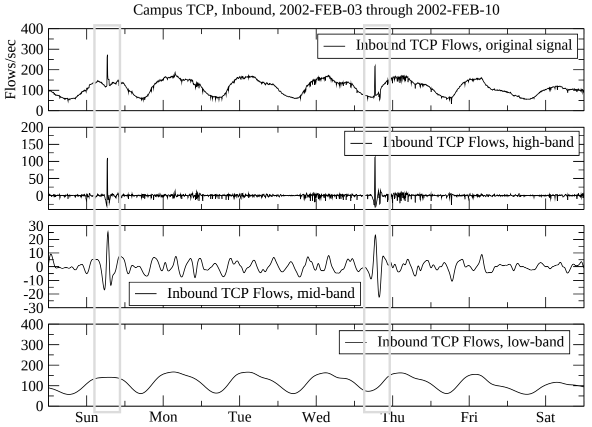

Fig. 5. Baseline signal of packet flows for a one week period highlighting two short-lived DoS attack anomalies plus high/mid/low decomposition.

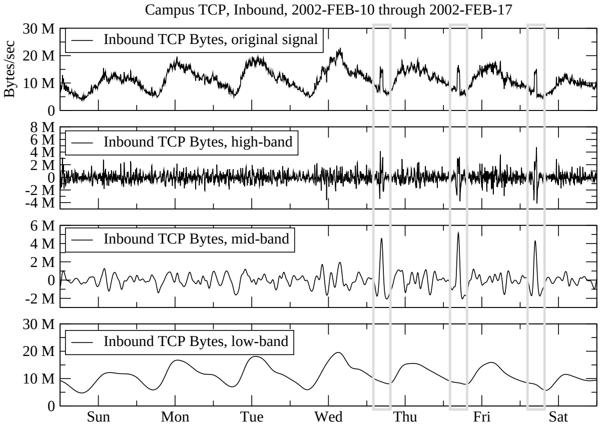

Fig. 6. Baseline signal of byte traffic from flow data for a one week period showing three short-lived measurement anomalies plus high/mid/low decomposition.

Another type of short-term anomaly is shown in Figure 6. This figure shows a periodic sequence of three measurement anomalies observed over a three day period. This was found to be a host in the outside world performing nightly backups to a campus backup server. The large volume of traffic each day was due to misconfiguration of the client backup software. As in the prior example, the decomposition easily isolates the anomaly signal in the high and mid bands while the low band is not affected by the anomaly. However if this anomaly had been intended behavior, accounting for it in high and mid bands would require additional filters in our analysis platform.

D. A Discriminator for Short-term Anomalies

One of the objectives of this work is to provide a basis for automating anomaly detection. It is important for any anomaly detection mechanism to minimize false positives and false negatives. Our analysis led to the development of the "deviation score" discrimination function for short-term anomalies described in Section IV.

Figure 7 shows how deviation scores can be used to highlight a series of short-term anomalies. The figure shows inbound TCP packet rate during a week plus three anomalies that might otherwise be difficult to discern from the baseline signal. Two of the anomalies are DoS floods that are easily detected and exposed automatically by their deviation scores and are marked by the first and second grey bands. Note, the bands are actually score values as shown by the scale on the right of the figure (the leftmost score does not quite reach 2). The third band marks an measurement anomaly unrelated to the DoS attacks.

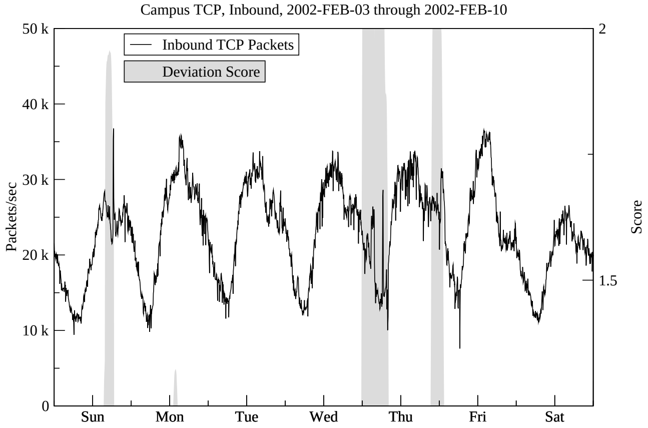

Fig. 7. Deviation analysis exposing two DoS attacks and one measurement anomaly in for a one week period in packet count data.

In Figure 8, we present a deviation analysis during a week containing a network outage. This outage affected about one fourth of the campus' IPv4 address space, and therefore caused an overall decrease in traffic. For each traffic measurement metric (packets, bytes, flows), inbound or outbound, our deviation scoring identified and marked the anomaly. This suggests that it is feasible to use a "rules based" approach or weighted average to determine the type or scope of the anomaly based on the accumulated impact of a set of deviation scores.

E. Exposing Anomalies in Aggregate Signals

An important issue in detecting traffic anomalies is the relationship between the strength of an anomaly's signal in a set of aggregated traffic. This is most easily considered with respect to the point at which measurement data is collected in a network. Intuition would say that an anomaly measured close to its source should be very evident while the same anomaly would be less evident if its signal were aggregated with a large amount of other traffic. We investigate this issue by isolating a specific subnet in which a system is the victim of a DoS attack.

Figure 9 shows the deviation analysis of two inbound DoS floods within the aggregate traffic of the victims 254 host subnet

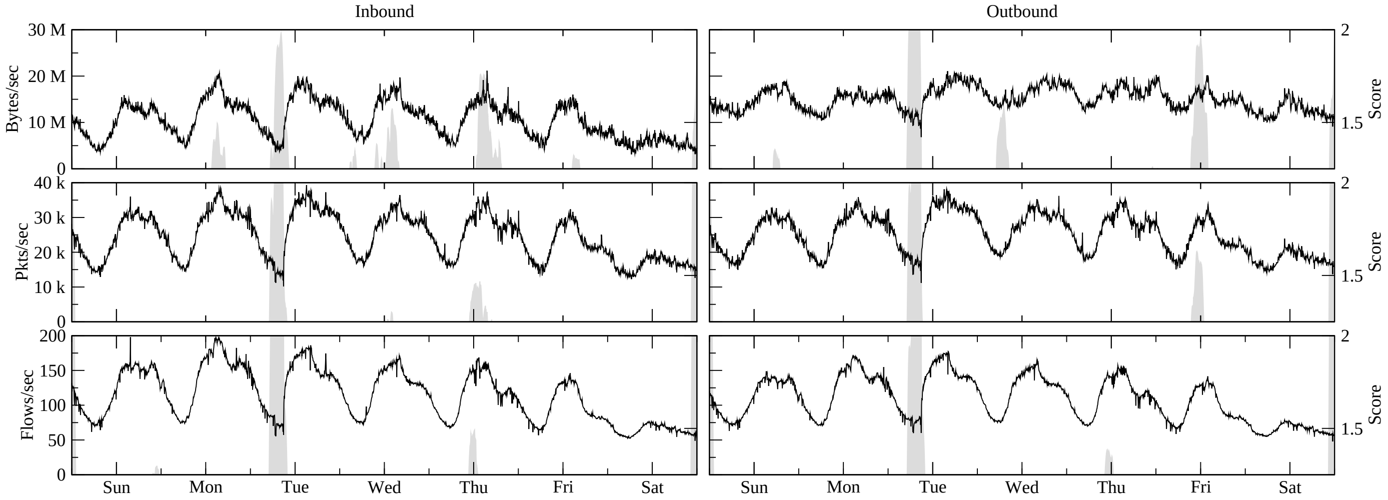

Fig. 8. Deviation analysis exposing a network outage of one (of four) Class-B networks.

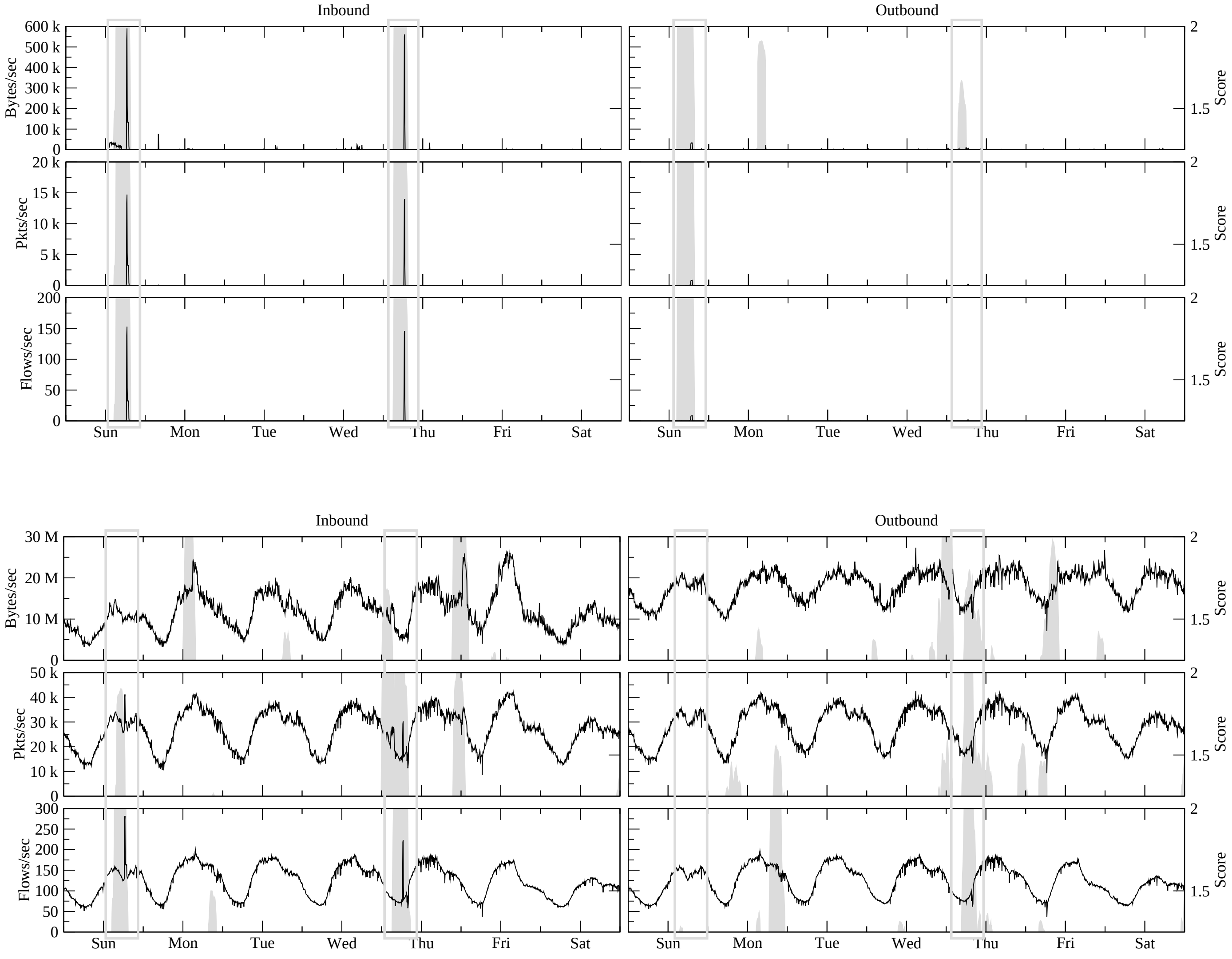

Fig. 9. Deviation analysis of two DoS events as seen in the 254 host subnet containing the victim of the attack (top) and in the aggregate traffi c of the entire campus.

and the aggregate traffic of the campus' four Class-B networks. The top set of graphs in this figure show that deviation scores easily highlight the extreme nature of DoS floods in inbound traffic within the subnet, and that they even highlight the much less evident outbound traffic anomaly. The bottom set of graphs show again that deviation scores highlight the same anomalies (as well as a number of others).

F. Hidden Anomalies

Through the application of our methods, we were able to identify a number of "hidden" anomalies in our data sets. These are anomalies that had not been previously identified by the campus network engineers. The majority of these were DoS attacks most of which could be identified by careful visual inspection.

One hidden anomaly of interest is shown in Figure 10. This figure shows outbound traffic from one of the campus' class-B networks during a four week period. The duration of this anomaly prevented its detection via deviation score. Decomposition enabled us to identify an anomaly that had previously gone unnoticed and was not easily seen visually. The anomaly is most visible following December 18th in the low-band graph where traffic remained uncharacteristically high across two subsequent days. Follow-up investigation using our repository of flow records showed this anomaly to have been due to of network abuse in which four campus hosts had their security compromised and were being remotely operated as peer-to-peer file servers.

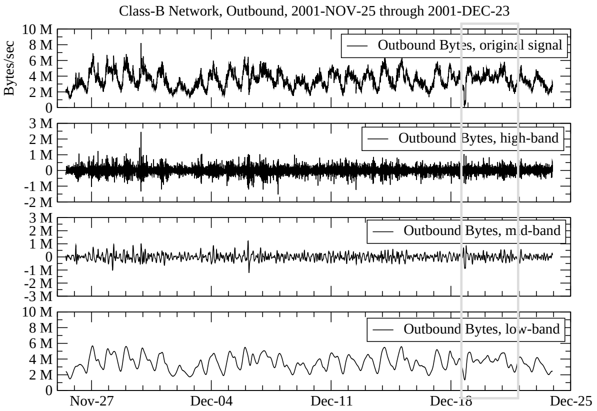

Fig. 10. Example of three-band analysis exposing a multi-day network abuse anomaly.

VI. Deviation Score Evaluation

We evaluated the results of our deviation scoring method of anomaly detection in two ways. First, we selected a set of anomalies logged in the network operator journal as a baseline and evaluated deviation score detection capability. Secondly, we used the same baseline set of anomalies to evaluate the effectiveness of an alternative detection technique based on Holt-Winters Forecasting [12] as a comparison to deviation scoring. We were limited in the extent to which we could evaluate either detection method since the baseline set of anomalies used in this analysis is unlikely to be complete. Therefore, we did not attempt to determine which method reported more false-positives.

TABLE II

Comparison of anomaly detection methods.

Total Candidate Anomalies Evaluated	Candidates detected by Deviation Score	Candidates detected by Holt-Winters
39	38	37

In each case we were somewhat tolerant of discrepancies between the anomaly timestamps in the journal's log entries and the times at which the automated methods reported anomalous network traffic. Specifically, we allowed a discrepancy of as much as 1.5 hours since both automated techniques sometimes shift the report significantly from the time of the event's onset. The respective identification of anomalies from our evaluation set is summarized in Table II. As can be seen, both techniques performed well in that their false-negative reports for the 39 anomalies in the candidate set were very low.

A. Deviation Score vs. Logged Anomalies

We selected 39 events from the network operator's log of anomalies. This subset of events were those for which a suitable amount of evidence had been gathered to label them as "high confidence" anomalies. This evidence gathering was a tedious effort often involving the examination of individual flow records to identify the specific IP header values for the packets that comprised the anomaly.

Of those 39 anomalies selected as a baseline for evaluation, deviation score analysis detected 38 of them with a significantly confident score of 1.7 or higher. For the single anomaly which wasn't detected by our method, its deviation score was substantial but less than 1.7. Visual inspection of the plot of this signal showed that this was due to a more prominent anomaly which was detected earlier in the week, which suppressed the magnitude of the undetected anomaly's score. This is a side-effect of the normalization within the context of the week-long window we used in our analysis method.

B. Holt-Winters Forecasting vs. Logged Anomalies

To further evaluate the deviation score, we compared its anomaly detection results with the Holt-Winters Forecasting and reporting technique that has been implemented in the freely-available development source code of RRDTOOL version 1.1. Holt-Winters Forecasting is a algorithm that builds upon exponential smoothing which is described in [27]. The specific implementation of Holt-Winters Forecasting is described in [12].

We selected this Holt-Winters method for comparison because it is perhaps the most sophisticated technique that is being used currently by network operators (although not yet by many!). The most common techniques in use employ simple site-specific rules-based thresholding. We summarily rejected those simple techniques because their rules and magic numbers for thresholding are often not portable beyond the local network and because of their general inability to handle seasonal effects such as daily cycles in signal amplitude. Both deviation score and Holt-Winters analysis can be configured to take a seasonal period into consideration, and both are being proposed as possible alternatives to analysis by visual inspection of network traffic signals.

As with our deviation score method, the Holt-Winters method also has parameters, and we configured it as follows. When constructing the RRDTOOL databases, the “HWPREDICT” Round-Robin Archive (RRA) was configured with alpha = 0.1, beta = 0.0035, and a seasonal period of 288, which is a period of one day at five minute intervals. The implementation’s default failure-threshold of 7 and a window-length of 9 were used. This means that a minimum of 7 violations (observed values outside the confidence bounds) within a window of 9 values was considered a high-confidence “anomaly”. Other HWPREDICT parameters had the default values specified by the implementation’s author.

The Holt-Winters method detected 37 out of the 39 anomalies in our evaluation set. By visual inspection of the anomalous signals plotted together with the deviation scores and Holt-Winters results (as shown in Figures 11 and 12), we have made the following observations. Two logged anomalies were not reported confidently by the Holt-Winters method. However, careful inspection of the Holt-Winters predictions for those anomalies showed that both would have been reported if only we had use a larger window size for the reporting phase. That is, the time at which Holt-Winters method reported the feature as anomalous lagged behind the original time at which the event was logged by the operator.

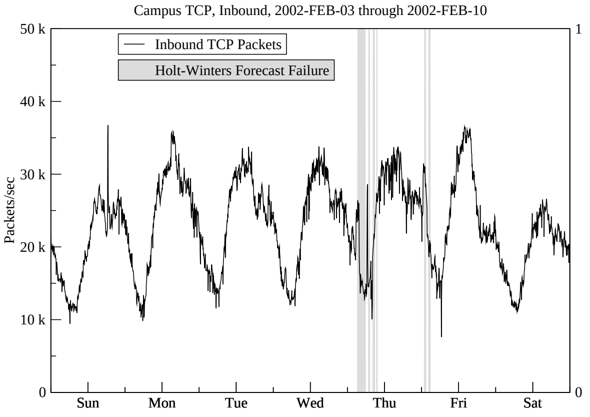

Fig. 11. Holt-Winters results corresponding to Figure 7

C. Deviation Scores vs. Holt-Winters Forecasting

We found the Holt-Winters method to be more sensitive to potential anomalies than our deviation score method. It reported many more “failure” (ostensibly anomalous) events (not included in our log) than did our method. These may have been false-positives. The reasons for these reports are perhaps two-fold. First, the Holt-Winters method’s parameters determine its sensitivity, in part, and it may have been poorly configured given for our input data set. Secondly, the deviation score technique tends to “blur” signal features by widening them with respect to time, making it less likely for a single anomaly to be erroneously reported multiple times. Also, the Holt-Winters boolean reporting sometimes oscillated between true and false within what appeared to be a single anomalous feature as determined by visual inspection. That is, the Holt-Winters method sometimes reported what was ostensibly a single anomaly as more than one discrete event. However, this behavior could be mitigated by changing the parameter values, so this is not a necessarily a general criticism of the Holt-Winters approach.

We also observed that our deviation score method more readily reported small amplitude features in the signal than did the Holt-Winters method. This is likely due to the Holt-Winters method requiring that the given feature’s amplitude to travel outside its “confidence band” of a particular width to be reported as an anomaly, whereas the deviation score method has no such fixed values with respect to signal amplitude. Instead the deviation score was based on normalized signal amplitudes within a window that spanned 2048 five-minute data points, or just over one week. Consequently, the deviation score values as currently configured are not necessarily applicable outside that week-long window. Rather, they are relative to the ambient or average traffic levels seen within that window. This means that some anomaly’s deviation score might be lowered by the presence of a more prominent anomaly within the same week. This is the same effect that happens with visual inspection if the vertical axis (*i.e.* the signal’s amplitude) is auto-scaled in a time-series plot. That is, a prominent anomaly will draw the observer’s attention away from a lesser one.

VII. Conclusions and Future Work

In this paper we present a signal analysis of network traffic anomalies in IP flow and SNMP data collected at the University of Wisconsin’s border router. Our data set spans 6 months and includes a catalog of over 100 anomalies which we organize into two distinct categories (short-lived events and long-lived events) for analysis.

We developed the IMAPIT environment to facilitate our work. This environment combines a flexible database system and robust signal analysis capability. We applied a variety of time-frequency analysis techniques to the data to determine how best to isolate anomalies. We found a wavelet system that effectively isolates both short and long-lived traffic anomalies. In addition to this system, we developed the concept of a *deviation score* which considers signal variation in both the high and medium frequency bands. We found this score to be extremely effective at isolating anomalies and to be very amenable for use in the generation of threshold-based alerts.

Evaluation of deviation scores as a means for automating anomaly detection shows it to be similarly effective to sophisticated time-series techniques such as Holt-Winters Forecasting. Both techniques have a small set of tunable parameters, and can perform poorly if configured incorrectly or perform well if their parameters are configured appropriately. For the set of 39 anomalies that we used for evaluation, both methods performed well in that their false-negative reports were negligible.

These results indicate that traffic anomaly detection mechanisms based on deviation score techniques may be effective, however further development is necessary. In the future, we plan to investigate machine learning methods to evaluate the impact of additional features in deviation scores. We also intend to investigate how well the deviation score method can be implemented to detect anomalies in real time. Furthermore we will study methods for *classifying* anomalies which would facilitate

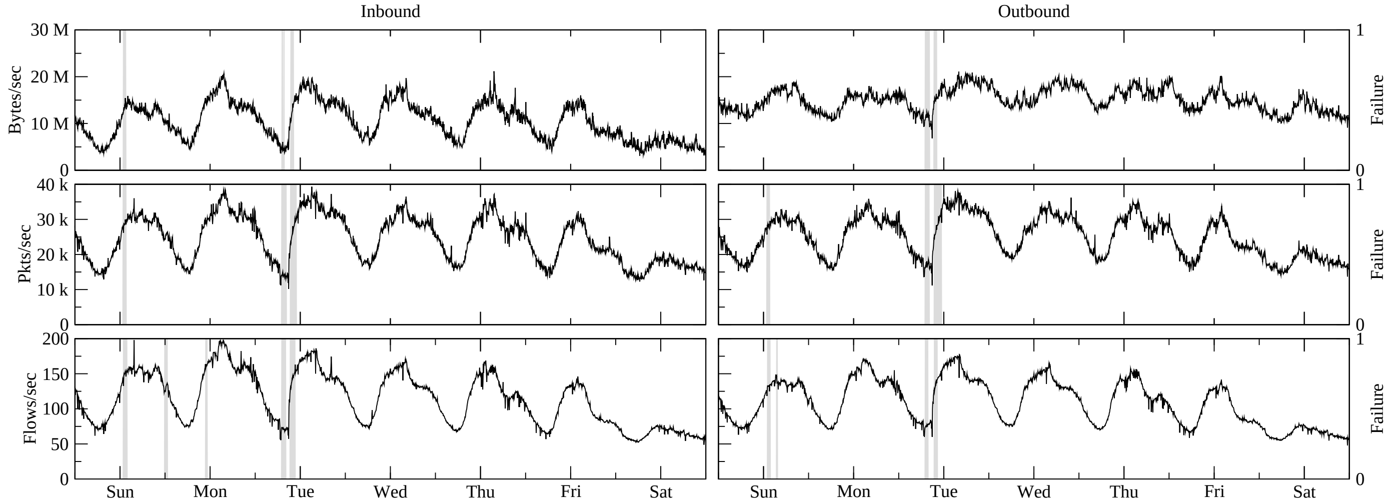

Fig. 12. Holt-Winters results corresponding to Figure 8

their diagnosis and treatment. Finally, we intend to pursue the idea of coordinated anomaly detection at multiple measurement locations in the wide area.

Acknowledgements

We wish to thank Darryl Veitch for his comments on this work. We also thank the anonymous reviewers for their helpful feedback.

References

[1] P. Barford and D. Plonka, "Characteristics of network traffic flow anomalies," in *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, November 2001.

[2] R. Cáceres, "Measurements of wide-area Internet traffic," Tech. Rep. UCB/CSD 89/550, Computer Science Department, University of California, Berkeley, 1989.

[3] V. Paxon, "Fast, approximate synthesis of fractional gaussian noise for generating self-similar network traffic," *Computer Communications Review*, vol. 27(5), pp. 5–18, October 1997.

[4] V. Paxon, *Measurements and Analysis of End-to-End Internet Dynamics*, Ph.D. thesis, University of California Berkeley, 1997.

[5] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson, "Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71–86, February 1997.

[6] P. Abry and D. Veitch, "Wavelet analysis of long range dependent traffic," *IEEE Transactions on Information Theory*, vol. 44, no. 1, 1998.

[7] K. Claffy, *Internet Traffic Characterization*, Ph.D. thesis, University of California, San Diego, 1994.

[8] C. Hood and C. Ji, "Proactive network fault detection," in *Proceedings of IEEE INFOCOM '97*, Kobe, Japan, April 1997.

[9] I. Katzela and M. Schwartz, "Schemes for fault identification in communications networks," *IEEE/ACM Transactions on Networking*, vol. 3(6), pp. 753–764, December 1995.

[10] A. Ward, P. Glynn, and K. Richardson, "Internet service performance failure detection," *Performance Evaluation Review*, August 1998.

[11] F. Feather, D. Siewiorek, and R. Maxion, "Fault detection in an ethernet network using anomaly signature matching," in *Proceedings of ACM SIGCOMM '93*, San Francisco, CA, September 2000.

[12] J. Brutlag, "Aberrant behavior detection in time series for network monitoring," in *Proceedings of the USENIX Fourteenth System Administration Conference LISA XIV*, New Orleans, LA, December 2000.

[13] J. Toelle and O. Niggemann, "Supporting intrusion detection by graph clustering and graph drawing," in *Proceedings of Third International Workshop on Recent Advances in Intrusion Detection RAID 2000*, Toulouse, France, October 2000.

[14] K. Fox, R. Henning, J. Reed, and R. Simonian, "A neural network approach towards intrusion detection," Tech. Rep., Harris Corporation, July 1990.

[15] N. Ye, "A markov chain model of temporal behavior for anomaly detection," in *Workshop on Information Assurance and Security*, West Point, NY, June 2000.

[16] D. Moore, G. Voelker, and S. Savage, "Inferring internet denial-of-service activity," in *Proceedings of 2001 USENIX Security Symposium*, Washington, DC, August 2001.

[17] V. Paxson, "Bro: A system for detecting network intruders in real-time," *Computer Networks*, vol. 31, no. 23-24, pp. 2435–2463, 1999.

[18] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites," in *WWW-02*, Hawaii, May 2002.

[19] R. Manajan, S. Bellovin, S. Floyd, V. Paxson, S. Shenker, and J. Ioannidis, "Controlling high bandwidth aggregates in the network," ACIRI Draft paper, February 2001.

[20] T. Oetiker, "The multi router traffic grapher," in *Proceedings of the USENIX Twelfth System Administration Conference LISA XII*, December 1998.

[21] K. McCloghrie and F. Kastenholz, "The interfaces group MIB," IETF RFC 2863, June 2000.

[22] S. Romig and M. Fullmer, "The OSU flow-tools package and cisco netflow logs," in *Proceedings of the USENIX Fourteenth System Administration Conference LISA XIV*, New Orleans, LA, December 2000.

[23] D. Plonka, "Flowscan: A network traffic flow reporting and visualization tool," in *Proceedings of the USENIX Fourteenth System Administration Conference LISA XIV*, New Orleans, LA, December 2000.

[24] Cisco IOS NetFlow, "http://www.cisco.com/go/netflow," 2002.

[25] I. Daubechies, B. Han, A. Ron, and Z. Shen, "Framelets: MRA-based constructions of wavelet frames," Preprint: ftp://ftp.cs.wisc.edu/Approx/dhrs.ps, 2001.

[26] Emmanuel Bacry, "Lastwave," $http : //www.cmap.polytechnique.fr/ \sim bacry/LastWave.$

[27] P. Brockwell and R. Davis, *Introduction to Time Series and Forecasting*, Springer, 1996.