Internet Traffic Behavior Profiling for Network Security Monitoring

Kuai Xu, Zhi-Li Zhang, *Member, IEEE*, and Supratik Bhattacharyya

***Abstract*—Recent spates of cyber-attacks and frequent emergence of applications affecting Internet traffic dynamics have made it imperative to develop effective techniques that can extract, and make sense of, significant communication patterns from Internet traffic data for use in network operations and security management. In this paper, we present a general methodology for building comprehensive behavior profiles of Internet backbone traffic in terms of communication patterns of end-hosts and services. Relying on data mining and entropy-based techniques, the methodology consists of significant cluster extraction, automatic behavior classification and structural modeling for in-depth interpretive analyses. We validate the methodology using data sets from the core of the Internet.**

***Index Terms*—Anomaly behavior, monitoring, traffic profiling.**

I. Introduction

As THE Internet continues to grow in size and complexity, the challenge of effectively provisioning, managing and securing it has become inextricably linked to a deep understanding of Internet traffic. Although there has been significant progress in instrumenting data collection systems for high-speed networks at the core of the Internet, developing a comprehensive understanding of the collected data remains a daunting task. This is due to the vast quantities of data, and the wide diversity of end-hosts, applications and services found in Internet traffic. While there exists an extensive body of prior work on traffic characterization on IP backbones—especially in terms of statistical properties (e.g., heavy-tail, self-similarity) for the purpose of network performance engineering, there has been very little attempt to build *general profiles* in terms of *behaviors*, i.e., communication patterns of end-hosts and services. The latter has become increasingly imperative and urgent in light of wide spread cyber attacks and the frequent emergence of disruptive applications that often rapidly alter the dynamics of network traffic, and sometimes bring down valuable Internet services. There is a pressing need for techniques that can extract underlying structures and significant communication patterns from Internet traffic data for use in network operations and security management.

The goal of this paper is to develop a general methodology for profiling Internet backbone traffic that 1) not only automatically discovers significant behaviors of interest from massive traffic data but 2) also provides a plausible interpretation of these behaviors to aid network operators in understanding and quickly identifying anomalous events with a significant amount of traffic, e.g., large scale scanning activities, worm outbreaks, and denial of service attacks. This second aspect of our methodology is both important and necessary due to the large number of interesting events and limited human resources. For these purposes, we employ a combination of data mining and entropy-based techniques to automatically cull useful information from largely unstructured data. We then classify and build structural models to characterize host/service behaviors of similar patterns (e.g., does a given source communicate with a single destination or with a multitude of destinations?).

In our study we use packet header traces collected on Internet backbone links in a tier-1 ISP, which are aggregated into *flows* based on the well-known five-tuple—the source IP address (`srcIP`), destination IP address (`dstIP`), source port (`srcPrt`), destination port (`dstPrt`), and protocol fields. Since our goal is to profile traffic in terms of communication patterns, we start with the essential four-dimensional feature space consisting of `srcIP`, `dstIP`, `srcPrt` and `dstPrt`. Using this four-dimensional feature space, we extract clusters of significance along each dimension, where each cluster consists of flows with the same feature value (referred to as *cluster key*) in the said dimension. This leads to four collections of interesting clusters—`srcIP` clusters, `dstIP` clusters, `srcPrt` clusters, and `dstPrt` clusters. The first two represent a collection of host behaviors while the last two represent a collection of service behaviors. In extracting clusters of significance, instead of using a fixed threshold based on volume, we adopt an entropy-based approach that culls interesting clusters based on the underlying feature value distribution (or *entropy*) in the fixed dimension. Intuitively, clusters with feature values (cluster keys) that are distinct in terms of distribution are considered significant and extracted; this process is repeated until the remaining clusters appear indistinguishable from each other. This yields a cluster extraction algorithm that automatically adapts to the traffic mix and the feature in consideration.

Given the extracted clusters along each dimension of the feature space, the second stage of our methodology is to discover "structures" among the clusters, and build common behavior models for traffic profiling. For this purpose, we first develop a behavior classification scheme based on observed similarities/dissimilarities in communication patterns. For every cluster, we compute an entropy-based measure of the variability or *uncer-*

Manuscript received March 25, 2006; revised March 31, 2007 and July 29, 2007. First published February 22, 2008; current version published December 17, 2008. Approved by IEEE/ACM Transactions on Networking Editor D. Veitch. This work was supported in part by the National Science Foundation (NSF) under Grants CNS-0435444 and CNS-0626812, in part by a University of Minnesota Digital Technology Center DTI grant, and in part by a Sprint ATL gift grant.

K. Xu is with Yahoo, Sunnyvale, CA 94089 USA (e-mail: kuai@yahoo-inc.com; kxu@cs.umn.edu).

Z.-L. Zhang is with Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: zhzhang@cs.umn.edu).

S. Bhattacharyya is with SnapTell Inc, Palo Alto, CA 94306 USA.

Digital Object Identifier 10.1109/TNET.2007.911438

tainty of each dimension except the (fixed) cluster key dimension, and use the resulting metrics to create *behavior classes*. We study the characteristics of these behavior classes over time as well as the dynamics of individual clusters, and demonstrate that the proposed classification scheme is robust and provides a natural basis for grouping together clusters of similar behavior patterns.

In the next step, we adopt ideas from structural modeling to develop the *dominant state analysis* technique for modeling and characterizing the interaction of features within a cluster. This leads to a compact "structural model" for each cluster based on *dominant states* that capture the most common or significant feature values and their interaction. The dominant state analysis serves two important purposes. First, it provides support for our behavior classification—we find that clusters within a behavior class have nearly identical forms of structural models. Second, it yields compact summaries of cluster information which provides interpretive value to network operators for explaining observed behavior, and may help in narrowing down the scope of a deeper investigation into specific clusters. In addition, we investigate additional features such as average flow sizes of clusters (in terms of both packet and byte counts) and their variabilities, and use them to further characterize similarities/dissimilarities among behavior classes and individual clusters.

We validate our approach using traffic data collected from a variety of links at the core of the Internet, and find that our approach indeed provides a robust and meaningful way of characterizing and interpreting cluster behavior. We show that several popular services and applications, as well as certain types of malicious activities, exhibit stable and distinctive behavior patterns in terms of the measures we formulate. The existence of such "typical" behavior patterns in traffic makes it possible to separate out a relatively small set of "atypical" clusters for further investigation. To this end, we present case studies highlighting a number of clusters with unusual characteristics that are identified by our profiling techniques, and demonstrate that these clusters exhibit malicious or unknown activities that are worth investigating further. Thus our technique can be a powerful tool for network operators and security analysts with applications to critical problems such as detecting anomalies or the spread of hitherto unknown security exploits, profiling unwanted traffic, tracking the growth of new services or applications, and so forth.

The contributions of this paper are summarized as follows.

- We present a novel adaptive threshold-based clustering approach for extracting significant clusters of interest based on the underlying traffic patterns.
- We introduce an entropy-based behavior classification scheme that automatically groups clusters into classes with distinct behavior patterns.
- We develop structural modeling techniques for interpretive analyses of cluster behaviors.
- Applying our methodology to Internet backbone traffic, we identify canonical behavior profiles for capturing typical and common communication patterns, and demonstrate how they can be used to detect interesting, anomalous or atypical behaviors.

The remainder of the paper is organized as follows. Section II provides some background. The adaptive-threshold clustering algorithm is presented in Section III. In Section IV we introduce the behavior classification and study its temporal characteristics. We present the dominant state analysis and additional feature exploration in Section V, and apply our methodology for traffic profiling in Section VI. Section VII discusses the related work. Section VIII concludes the paper.

II. Background and Datasets

Information essentially quantifies "the amount of uncertainty" contained in data [1]. Consider a random variable X that may take N_X discrete values. Suppose we randomly sample or observe X for m times, which induces an empirical probability distribution[1] on X, $p(x_i) = m_i/m, x_i \in X$, where m_i is the frequency or number of times we observe X taking the value x_i. The (empirical) *entropy* of X is then defined as

$$H(X) := -\sum_{x_i \in X} p(x_i) \log p(x_i) \tag{1}$$

where by convention $0 \log 0 = 0$.

Entropy measures the "observational variety" in the observed values of X [2]. Note that unobserved possibilities (due to $0 \log 0 = 0$) do not enter the measure, and $0 \le H(X) \le H_{max}(X) := \log \min\{N_X, m\}$. $H_{max}(X)$ is often referred to as the *maximum entropy* of (sampled) X, as $2^{H_{max}(X)}$ is the maximum number of possible *unique* values (i.e., "maximum uncertainty") that the observed X can take in m observations. Clearly $H(X)$ is a function of the support size N_X and sample size m. Assuming that $m \ge 2$ and $N_X \ge 2$ (otherwise there is no "observational variety" to speak of), we define the *standardized* entropy below—referred to as *relative uncertainty* (RU) in this paper, as it provides an index of variety or uniformity regardless of the support or sample size

$$RU(X) := \frac{H(X)}{H_{max}(X)} = \frac{H(X)}{\log \min\{N_X, m\}}. \tag{2}$$

Clearly, if $RU(X) = 0$, then all observations of X are of the same kind, i.e., $p(x) = 1$ for some $x \in X$; thus observational variety is completely absent. More generally, let A denote the (sub)set of observed values in X, i.e., $p(x_i) > 0$ for $x_i \in A$. Suppose $m \le N_X$. Then $RU(X) = 1$ if and only if $|A| = m$ and $p(x_i) = 1/m$ for each $x_i \in A$. In other words, all observed values of X are different or *unique*, thus the observations have the highest degree of variety or uncertainty. Hence when $m \le N_X$, $RU(X)$ provides a measure of "randomness" or "uniqueness" of the values that the observed X may take—this is what is mostly used in this paper, as in general $m \ll N_X$.

In the case of $m > N_X$, $RU(X) = 1$ if and only if $m_i = m/N_X$, thus $p(x_i) = 1/N_X$ for $x_i \in A = X$, i.e., the observed values are *uniformly* distributed over X. In this case, $RU(X)$ measures the degree of uniformity in the observed values of X. As a general measure of *uniformity* in the observed values of X, we consider the conditional entropy $H(X|A)$ and *conditional relative uncertainty* $RU(X|A)$ by conditioning X based on A. Then we have $H(X|A) = H(X)$, $H_{max}(X|A) = \log|A|$ and $RU(X|A) = H(X)/log|A|$. Hence $RU(X|A) = 1$ if and only if $p(x_i) = 1/|A|$ for every $x_i \in A$. In general, $RU(X|A) \approx 1$

[1]With $m \to \infty$, the induced empirical distribution approaches the true distribution of X.

TABLE I
MULTIPLE LINKS USED IN OUR ANALYSIS

Link	Time	Utilization	Duration	Packets	Trace size
L_1	01/28/2004	78 Mbps	24 hours	$1.60 * 10^9$	95 GB
L_2	01/28/2004	86 Mbps	24 hours	$1.65 * 10^9$	98 GB
L_3	02/06/2004	40 Mbps	3 hours	$2.03 * 10^8$	12 GB
L_4	02/06/2004	52 Mbps	3 hours	$1.91 * 10^8$	11 GB
L_5	04/07/2003	207 Mbps	3 hours	$5.18 * 10^8$	28 GB

means that the observed values of X are closer to being uniformly distributed, thus less distinguishable from each other, whereas $RU(X|A) \ll 1$ indicates that the distribution is more skewed, with a few values more frequently observed. This measure of uniformity is used in Section III for defining "significant clusters of interest."

We conclude this section by providing a quick description of the datasets used in our study. The datasets consist of packet header (the first 44 bytes of each packet) traces collected from multiple links in a large ISP network at the core of the Internet (Table I). For every 5-minute time slot, we aggregate packet header traces into *flows*, which is defined based on the well-known 5-tuple (i.e., the source IP address, destination IP address, source port number, destination port number, and protocol) with a timeout value of 60 seconds [3]. The 5-minute time slot is used as a trade-off between timeliness of traffic behavior profiling and the amount of data to be processed in each slot.

III. EXTRACTING SIGNIFICANT CLUSTERS

We start by focusing on each dimension of the four-feature space, `srcIP`, `dstIP`, `srcPrt`, or `dstPrt`, and extract "significant clusters of interest" along this dimension. The extracted `srcIP` and `dstIP` clusters yield a set of "interesting" host behaviors (communication patterns), while the `srcPrt` and `dstPrt` clusters yield a set of "interesting" service/port behaviors, reflecting the aggregate behaviors of individual hosts on the corresponding ports. In the following we introduce our definition of *significance* using the (conditional) relative uncertainty measure.

Given one feature dimension $\boldsymbol{X}$ and a time interval T, let m be the total number of flows observed during the time interval, and $A = \{a_1, \ldots, a_n\}, n \geq 2$, be the set of distinct values (e.g., `srcIP`'s) in X that the observed flows take. Then the (induced) probability distribution $\mathcal{P}_A$ on X is given by $p_i := \mathcal{P}_A(a_i) = m_i/m$, where m_i is the number of flows that take the value a_i (e.g., having the `srcIP` a_i). Then the (conditional) relative uncertainty, $RU(\mathcal{P}_A) := RU(X|A)$, measures the degree of uniformity in the observed features A. Let β represent a large value close to 1, say, 0.9. If $RU(\mathcal{P}_A)$ is larger than β, then the observed values are close to being uniformly distributed, and thus *nearly indistinguishable*. Otherwise, there are likely feature values in A that "stand out" from the rest. We say a subset S of A contains the *most significant* (thus "interesting") values of A if S is the smallest subset of A such that i) the probability of any value in S is larger than those of the remaining values; and ii) the (conditional) probability distribution on the set of the remaining values, $R := A - S$, is close to being uniformly distributed, i.e., $RU(\mathcal{P}_R) := RU(X|R) > \beta$. Intuitively, S contains the most significant feature values in A, while the remaining values are nearly indistinguishable from each other.

To see what S contains, order the feature values of A based on their probabilities: let $\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_n$ be such that $\mathcal{P}_A(\hat{a}_1) \geq \mathcal{P}_A(\hat{a}_2) \geq \cdots \mathcal{P}_A(\hat{a}_n)$. Then $S\{\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{k-1}\}$ and $R = A - S = \{\hat{a}_k, \hat{a}_{k+1}, \ldots, \hat{a}_n\}$ where k is the smallest integer such that $RU(\mathcal{P}_R) > \beta$. Let $\alpha^* = \hat{a}_{k-1}$. Then α^* is the largest "cut-off" threshold such that the (conditional) probability distribution on the set of remaining values R is close to being uniformly distributed. To extract S from A (thereby, the clusters of flows associated with the significant feature values), we take advantage of the fact that in practice only a relatively few values (with respect to n) have significant larger probabilities, i.e., $|S|$ is relatively small, while the remaining feature values are close to being uniformly distributed. Hence we can efficiently search for the optimal cut-off threshold α^*.

Algorithm 1 Entropy-based Significant Cluster Extraction

```
1: Parameters: $\alpha := \alpha_0$; $\beta := 0.9$; $S := \emptyset$;
2: Initialization: $S := \emptyset$; $R := A$; $k := 0$;
3: compute prob. dist. $\mathcal{P}_R$ and its RU $\theta := RU(\mathcal{P}_R)$;
4: while $\theta \leq \beta$ do
5:    $\alpha = \alpha \times 2^{-k}$; $k++$;
6:    for each $a_i \in R$ do
7:       if $\mathcal{P}_A(a_i) \geq \alpha$ then
8:          $S := S \cup \{a_i\}$; $R := R - \{a_i\}$;
9:       end if
10:   end for
11:   compute (cond.) prob. dist. $\mathcal{P}_R$ and $\theta := RU(\mathcal{P}_R)$;
12: end while
```

Algorithm 1 presents an efficient *approximation* algorithm[2] (in pseudo-code) for extracting the significant clusters in S from A (thereby, the clusters of flows associated with the significant feature values). The algorithm starts with an appropriate initial value α_0 (e.g., $\alpha_0 = 2\%$), and searches for the optimal cut-off threshold α^* from above via "exponential approximation" (reducing the threshold α by an exponentially decreasing factor $1/2^k$ at the kth step). As long as the relative uncertainty of the (conditional) probability distribution $\mathcal{P}_R$ on the (remaining) feature set R is less than β, the algorithm examines each feature value in R and includes those whose probabilities exceed the threshold α into the set S of significant feature values. The algorithm stops when the probability distribution of the remaining feature values is close to being uniformly distributed (> a large value of β). Let $\hat{\alpha}^*$ be the final cut-off threshold (an approximation to α^*) obtained by the algorithm.

Fig. 1 shows the results we obtain by applying the algorithm to the 24-hour packet trace collected on L_1, where the significant clusters are extracted in every 5-minute time slot along `srcIP` and `dstIP` feature dimensions. In Fig. 1(a)–(b) we plot both the total number of distinct feature values as well as the number of significant clusters extracted in each 5-minute slot

[2]An efficient algorithm using binary search is also devised, but not used here.

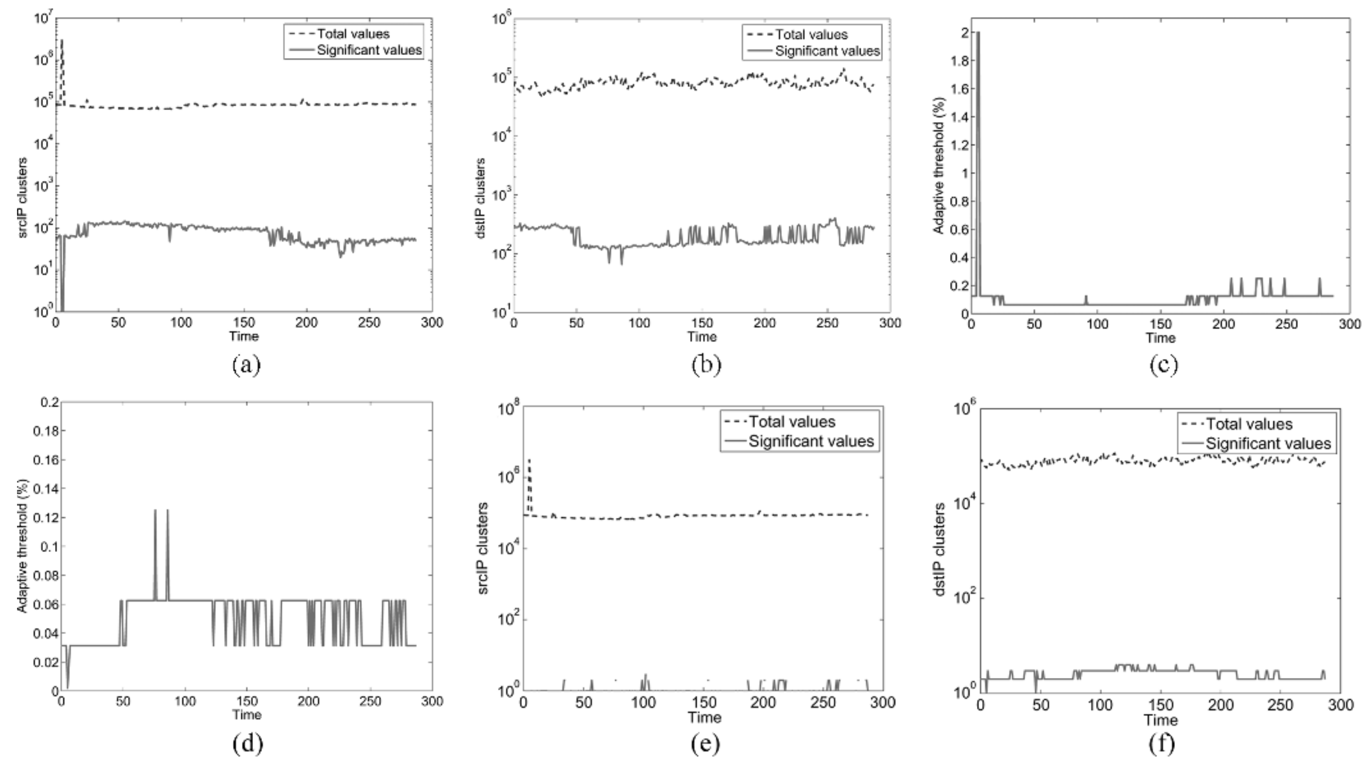

Fig. 1. Total number of distinct values and significant clusters extracted from `srcIP` and `dstIP` dimensions of L_1 over a one-day period (a)–(b) based on entropy-based adaptive thresholding algorithm. (c)–(d) Corresponding final cut-off threshold obtained by the entropy-based significant cluster extraction algorithm. (e)–(f) Total number of distinct values and significant clusters extracted from `srcIP` and `dstIP` dimensions using the algorithm in [4]. (a) Significant clusters of `srcIP` dimension. (b) Significant clusters of `dstIP` dimension. (c) Cut-off threshold of `srcIP` dimension. (d) Cut-off threshold of `dstIP` dimension. (e) Significant clusters of `srcIP` dimension using [4]. (f) Significant clusters of `dstIP` dimension using [4].

over 24 hours for `srcIP` and `dstIP` dimensions (note that the y-axis is in log scale). In Fig. 1(c)–(d), we plot the corresponding final cut-off threshold obtained by the algorithm. For both dimensions, the number of significant clusters is far smaller than the number of feature values n, and the cut-off thresholds for the different feature dimensions also differ. This shows that *no* single *fixed* threshold would be adequate in the definition of significant behavior clusters.

We see that while the total number of distinct values along a given dimension may not fluctuate very much, the number of significant feature values (clusters) may vary dramatically, due to changes in the underlying feature value distributions. These changes result in different cut-off thresholds being used in extracting the significant feature values (clusters). In fact, the dramatic changes in the number of significant clusters (or equivalently, the cut-off threshold) also signifies major changes in the underlying traffic patterns. Similar observations also hold for the `srcPrt` and `dstPrt` feature dimensions [5].

To compare our approach of finding significant clusters with existing techniques based on fixed threshold, we run the software package developed in [4] on the same packet traces. The package provides choices of four fixed thresholds, 2%, 5%, 10%, and 20%, and we select the lowest threshold 2% in our experiment. Fig. 1(e)–(f) show the number of total clusters and significant clusters for `srcIP` and `dstIP` dimensions, respectively. For both dimensions, we obtain a few clusters during each time period, which indicates the challenges for fixed threshold approaches to predict the "right" thresholds.

IV. Cluster Behavior Classification

In this section we introduce an *entropy-based* approach to characterize the "behavior" of the significant clusters extracted using the algorithm in the previous section. We show that this leads to a natural behavior classification scheme that groups the clusters into classes with distinct behavior patterns.

A. Behavior Class Definition

Consider the set of, say, `srcIP`, clusters extracted from flows observed in a given time slot. The flows in each cluster share the same cluster *key*, i.e., the same `srcIP` address, while they can take any possible value along the other three free dimensions, i.e., four basic dimensions except the cluster dimension. In this case, `dstIP`, `srcPort`, and `dstPort` are free dimensions. Hence the flows in a cluster induce a probability distribution on each of the three "free" dimensions, and thus a *relative uncertainty* (cf. Section II) measure can be defined. For each cluster extracted along a fixed dimension, we use X, Y and Z to denote its three "free" dimensions, using the convention listed in Table II. Hence for a `srcIP` cluster, X, Y, and Z denote the `srcPrt`, `dstPrt` and `dstIP` dimensions, respectively. This cluster can be characterized by an *RU vector* $[RU_X, RU_Y, RU_Z]$.

In Fig. 2 we represent the RU vector of each `srcIP` cluster extracted in each 5-minute time slot over a 1-hour period from L_1 as a point in a unit-length cube. We see that most points are "clustered" (in particular, along the axes), suggesting that there

TABLE II
CONVENTION OF FREE DIMENSION DENOTATIONS

Cluster key	Free dimensions		
	X	Y	Z
srcIP	srcPrt	dstPrt	dstIP
dstIP	srcPrt	dstPrt	srcIP
srcPrt	dstPrt	srcIP	dstIP
dstPrt	srcPrt	srcIP	dstIP

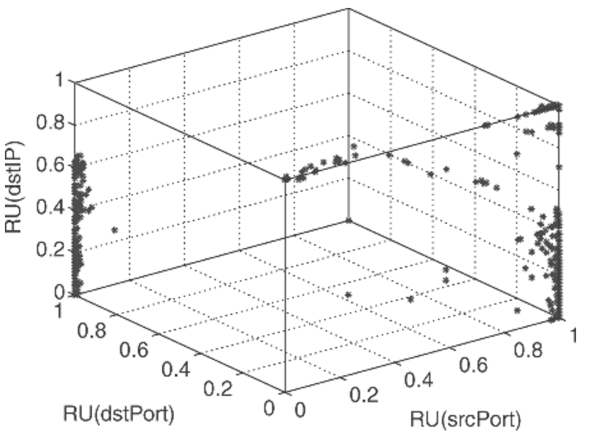

Fig. 2. Distribution of RU vectors for `srcIP` clusters from L_1 during a 1-hour period.

are certain common "behavior patterns" among them. Similar results using the `srcIP` clusters on four other links are also presented in [5]. This "clustering" effect can be explained by the "multi-modal" distribution of the relative uncertainty metrics along each of the three free dimensions of the clusters, as shown in Fig. 3(a)–(c) where we plot the histogram (with a bin size of 0.1) of RU_X, RU_Y and RU_Z of all the clusters on links L_1 to L_5 respectively. For each free dimension, the RU distribution of the clusters is multi-modal, with two strong modes (in particular, in the case of `srcPrt` and `dstPrt`) residing near the two ends, 0 and 1. Similar observations also hold for `dstIP`, `srcPrt` and `dstPrt` clusters extracted on these links.

As a *convenient* way to group together clusters of similar behaviors, we divide each RU dimension into three categories (assigned with a label): 0 (low), 1 (medium) and 2 (high), using the following criteria:

$$L(ru) = \begin{cases} 0\,(low), & \text{if } 0 \le ru \le \epsilon \\ 1\,(medium), & \text{if } \epsilon < ru < 1-\epsilon \\ 2\,(high), & \text{if } 1-\epsilon \le ru \le 1 \end{cases} \tag{3}$$

where for the `srcPrt` and `dstPrt` dimensions, we choose $\epsilon = 0.2$, while for the `srcIP` and `dstIP` dimensions, $\epsilon = 0.3$. This labelling process classifies clusters into 27 possible *behavior classes* (*BC* in short), each represented by a (label) vector $[L(RU_X), L(RU_Y), L(RU_Z)] \in \{0,1,2\}^3$. For ease of reference, we also treat $[L(RU_X), L(RU_Y), L(RU_Z)]$ as an integer (in ternary representation) $id = L(RU_X)\cdot 3^2 + L(RU_Y)\cdot 3 + L(RU_Z) \in \{0,1,2,\ldots,26\}$, and refer to it as BC_{id}. Hence `srcIP` $BC_6 = [0,2,0]$, which intuitively characterizes the communicating behavior of a host using a single or a few `srcPrt`'s to talk with a single or a few `dstIP`'s on a larger number of `dstPrt`'s. We remark here that for clusters extracted using other *fixed* feature dimensions (e.g., `srcPrt`, `dstPrt` or `dstIP`), the BC labels and id's have a different meaning and interpretation, as the free dimensions are different (see Table II). We will explicitly refer to the BCs defined along each dimension as `srcIP` BCs, `dstIP` BCs, `srcPrt` BCs and `dstPrt` BCs. However, when there is no confusion, we will drop the prefix.

B. Temporal Properties of Behavior Classes

We now study the temporal properties of the behavior classes. We introduce three metrics to capture three different aspects of the characteristics of the BC's over time: 1) *popularity*: which is the number of times we observe a particular BC appearing (i.e., at least one cluster belonging to the BC is observed); 2) *(average) size*: which is the average number of clusters belonging to a given BC, whenever it is observed; and 3) *(membership) volatility*: which measures whether a given BC tends to contain the same clusters over time (i.e., the member clusters re-appear over time), or new clusters.

Formally, consider an observation period of T time slots. For each BC_i, let C_{ij} be the number of observed clusters that belong to BC_i in the time slot τ_j, $j = 1, 2, \ldots, T, O_i$ the number of time slots that BC_i is observed, i.e., $O_i = |\{C_{ij} : C_{ij} > 0\}|$, and U_i be the number of *unique clusters* belonging to BC_i over the entire observation period. Then the popularity of BC_i is defined as $\Pi_i = O_i/T$; its average size $\Sigma_i = \sum_{j=1}^{T} C_{ij}/O_i$; and its (membership) volatility $\Psi_i = U_i/\sum_{j=1}^{T} C_{ij} = U_i/(\Sigma_i O_i)$. If a BC contains the same clusters in all time slots, i.e., $U_i = C_{ij}$, for every j such that $C_{ij} > 0$, then $\Psi_i = 1/T$ and $\Psi_i \approx 0$ when T is large. In general, the closer Ψ_i is to 0, the less volatile the BC is. Note that the membership volatility metric is defined only for BC's with relatively high frequency, e.g., $\Pi > 0.2$, as otherwise it contains too few "samples" to be meaningful.

In Fig. 4(a)–(c) we plot Π_i, Σ_i and Ψ_i of the `srcIP` BC's for the `srcIP` clusters extracted using link L_1 over a 24-hour period, where each time slot is a 5-minute interval (i.e., $T = 288$). From Fig. 4(a) we see that 7 BC's, $BC_2[0,0,2]$, $BC_6[0,2,0]$, $BC_7[0,2,1]$, $BC_8[0,2,2]$, $BC_{18}[2,0,0]$, $BC_{19}[2,0,1]$ and $BC_{20}[2,0,2]$, are most popular, occurring more than half of the time; while $BC_{11}[2,0,2]$ and $BC_{12}[2,1,0]$ and $BC_{24}[2,2,1]$ have moderate popularity, occurring about one-third of the time. The remaining BC's are either rare or not observed at all. Fig. 4(b) shows that the five popular BC's, BC_2, BC_6, BC_7, BC_{18}, and BC_{20}, have the largest (average) size, each having around 10 or more clusters; while the other two popular BC's, BC_8 and BC_{19}, have four or fewer BC's on the average. The less popular BC's are all small, having at most one or two clusters on the average when they are observed. From Fig. 4(c), we see that the two popular BC_2 and BC_{20} (and the less popular BC_{11}, BC_{12} and BC_{24}) are most volatile, while the other five popular BC's, BC_6, BC_7, BC_8, BC_{18} and BC_{19} are much less volatile. To better illustrate the difference in the membership volatility of the 7 popular BC's, in Fig. 4(d) we plot U_i as a function of time, i.e., $U_i(t)$ is the total number of unique clusters belonging to BC_i *up to time slot* t. We see that for BC_2 and BC_{20}, new clusters show up in nearly every time slot, while for BC_7, BC_8 and BC_{19}, the same clusters re-appear again and again. For BC_6 and BC_{18}, new clusters show up gradually over time and they tend to re-occur, as evidenced

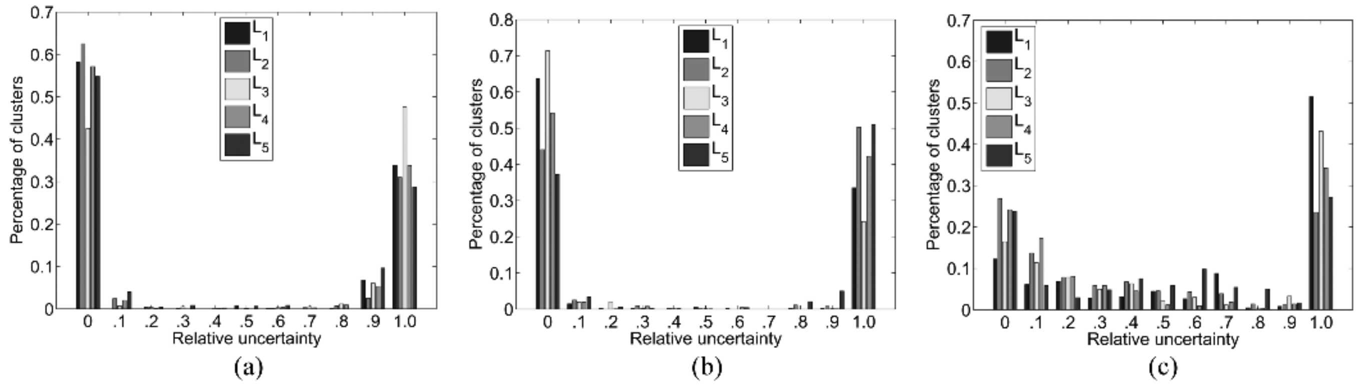

Fig. 3. Histogram distributions of relative uncertainty on free dimensions for `srcIP` clusters from L_{1-5} during a 1-hour period. (a) srcPrt free dimension; (b) dstPrt free dimension; (c) dstIP free dimension.

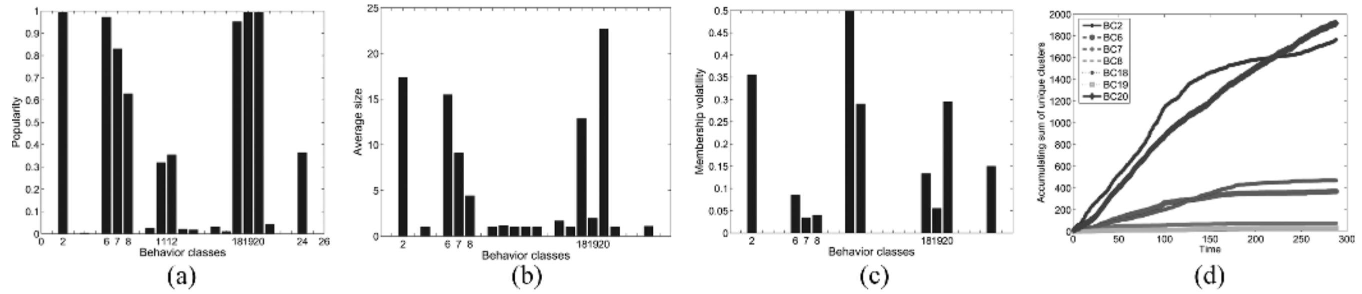

Fig. 4. Temporal properties of `srcIP` BCs using srcIP clusters on L_1 over a 24-hour period. (a) Popularity (Π). (b) Average size (Σ). (c) Volatility (Ψ). (d) $U_i(t)$ over time.

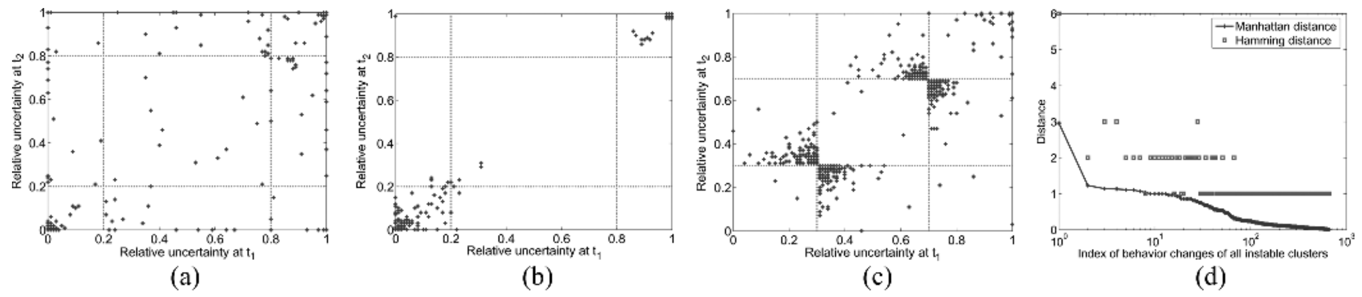

Fig. 5. Behavior transitions along `srcPrt`, `dstPrt` and `dstIP` dimensions as well as Manhattan and Hamming distances for "multi-BC" `srcIP` clusters on L_1. (a) `srcPrt` dimension. (b) `dstPrt` dimension. (c) `dstIP` dimension. (d) Transitions in d_m and d_h.

by the tapering off of the curves and the large average size of these two BC's.

C. Behavior Dynamics of Individual Clusters

We now investigate the behavior characteristics of individual clusters over time. In particular, we are interested in understanding i) the relation between the *frequency* of a cluster (i.e., how often it is observed) and the behavior class(es) it appears in; and ii) the behavior *stability* of a cluster if it appears multiple times, namely, whether a cluster tends to re-appear in the same BC or different BC's?

We use the set of `srcIP` clusters extracted on links with the longest duration, L_1 and L_2, over a 24-hour period as two representative examples to illustrate our findings. As shown in [5], the frequency distribution of clusters is "heavy-tailed": for example more than 90.3% (and 89.6%) clusters in L_1 (and L_2) occur fewer than 10 times, of which 47.1% (and 55.5%) occur only once; 0.6% (and 1.2%) occur more than 100 times. Next, for those clusters that appear at least twice (2443 and 4639 `srcIP` clusters from link L_1 and L_2, respectively), we investigate whether they tend to re-appear in the same BC or different BC's. We find that a predominant majority (nearly 95% on L_1 and 96% on L_2) stay in the same BC when they re-appear. Only a few (117 clusters on L_1 and 337 on L_2) appear in more than 1 BC. For instance, out of the 117 clusters on L_1, 104 appear in 2 BC's, 11 in 3 BC's and 1 in 5 BC's. We refer to these clusters as "multi-BC" clusters.

In Fig. 5(a)–(c) we examine the behavior transitions of those 117 "multi-BC" clusters on L_1 along each of the three dimensions (`srcPrt`, `dstPrt` and `dstIP`), where each point represents an RU transition $(RU(t_1), RU(t_2))$ in the corresponding dimension. We see that for each dimension, most of the points center around the diagonal, indicating that the RU values typically do not change significantly. For those transitions that cross the boundaries, causing a BC change for the corresponding cluster, most fall into the rectangle boxes along the sides, with only a few falling into the two square boxes on the upper left and lower right corners. This means that along each dimension, most of the BC changes can be attributed to transitions between two adjacent labels.

To measure the combined effect of the three RU dimensions on behavior transitions, we define two distance metrics: *Manhattan distance* (d_m) and *Hamming distance* (d_h)

$$d_m = |RU_X(t_1) - RU_X(t_2)| + |RU_Y(t_1) - RU_Y(t_2)| + |RU_Z(t_1) - RU_Z(t_2)| \quad (4)$$

and

$$d_h = |L_X(t_1) - L_X(t_2)| + |L_Y(t_1) - L_Y(t_2)| + |L_Y(t_1) - L_Y(t_2)| \quad (5)$$

where L is the labeling function [c.f., (3)].

Fig. 5(d) plots the Manhattan distance and Hamming distance of those behavior transitions that cause a BC change (a total of 658 such instances) for one of the "multi-BC" clusters. These behavior transitions are indexed in the decreasing order of Manhattan distance. We see that over 90% of the "BC-changing" behavior transitions have only a small Manhattan distance (e.g., $\leq$0.4), and most of the BC changes are within *akin* BC's, i.e., with a Hamming distance of 1. Only 60 transitions have a Manhattan distance larger than 0.4, and 31 have a Hamming distance of 2 or 3, causing BC changes between *non-akin* BC's. Hence, in a sense, only these behavior transitions reflect a large deviation from the norm. These "deviant" behavior transitions can be attributed to large RU changes in the `srcPrt` dimension, followed by the `dstIP` dimension. Out of the 117 multi-BC clusters, we find that only 28 exhibit one or more "deviant" behavior transitions (i.e., with $d_m \geq 0.4$ or $d_h = 2,3$) due to significant traffic pattern changes, and thus are regarded as *unstable* clusters. The above analysis has therefore enabled us to distinguish between this small set of clusters from the rest of the multi-BC clusters for which behavior transitions are between *akin* BCs, and a consequence of the choice of epsilon in (3), rather than any significant behavioral changes.

We conclude this section by commenting that our observations and results regarding the temporal properties of behavior classes and behavior dynamics of individual clusters hold not only for the `srcIP` clusters extracted on L_1 but also on other dimensions and links we studied. Such results are included in [5]. In summary, our results demonstrate that the behavior classes defined by our RU-based behavior classification scheme manifest *distinct* temporal characteristics, as captured by the frequency, populousness and volatility metrics. In addition, clusters (especially those frequent ones) in general evince consistent behaviors over time, with only a very few occasionally displaying unstable behaviors. In a nutshell, our RU-based behavior classification scheme inherently captures certain behavior similarity among (significant) clusters. This similarity is in essence measured by how varied (e.g., random or deterministic) the flows in a cluster assume feature values in the other three free dimensions. The resulting behavior classification is consistent and robust over time, capturing clusters with similar temporal characteristics.

V. Structural Models

In this section we introduce the *dominant state analysis* technique for modeling and characterizing the interaction of features within a cluster. We also investigate additional features, such as average flow sizes of clusters and their variabilities for further characterizing similarities/dissimilarities among behavior classes and individual clusters. The dominant state analysis and additional feature inspection together provide plausible interpretation of cluster behavior.

A. Dominant State Analysis

Our dominant state analysis borrows ideas from structural modeling or reconstructability analysis in system theory ([6]–[8]) as well as more recent graphical models in statistical learning theory [9]. The intuition behind our dominant state analysis is described below. Given a cluster, say a `srcIP` cluster, all flows in the cluster can be represented as a 4-tuple (ignoring the protocol field) $\langle u, x_i, y_i, z_i \rangle$, where the `srcIP` has a fixed value u, while the `srcPrt` (X dimension), `dsrPrt` (Y dimension) and `dstIP` (Z dimension) may take any legitimate values. Hence each flow in the cluster imposes a "constraint" on the three "free" dimensions X, Y and Z. Treating each dimension as a random variable, the flows in the cluster constrain how the random variables X, Y and Z "interact" or "depend" on each other, via the (induced) *joint* probability distribution $\mathcal{P}(X, Y, Z)$. The objective of dominant state analysis is to explore the interaction or dependence among the free dimensions by identifying "simpler" subsets of values or constraints (called *structural models* in the literature [6]) to represent or approximate the original data in their probability distribution. We refer to these subsets as *dominant states* of a cluster. Hence given the information about the dominant states, we can reproduce the original distribution with reasonable accuracy.

We use some examples to illustrate the basic ideas and usefulness of dominant state analysis. Suppose we have a `srcIP` cluster consisting mostly of scans (with a fixed `srcPrt` 220) to a large number of random destinations on `dstPrt` 6129. Then the values in the `srcPrt`, `dstPrt` and `dstIP` dimensions these flows take are of the form $\langle 220, 6129, * \rangle$, where $*$ (wildcard) indicates random or arbitrary values. Clearly this cluster belongs to `srcIP` BC_2 $[0, 0, 2]$, and the cluster is dominated by the flows of the form $\langle 220, 6129, * \rangle$. Hence the dominant state of the cluster is $\langle 220, 6129, * \rangle$, which approximately represents the nature of the flows in the cluster, even though there might be a small fraction of flows with other states. As a slightly more complicated example, consider a `srcIP` cluster which consists mostly of scanning traffic from the source (with randomly selected `srcPrt`) to a large number of random destinations on either `dstPrt` 139 (50% of the flows) or 445 (45%). Then the dominant states of the cluster (belonging to BC_{20}) are $\{\langle *, 139, * \rangle[50\%], \langle *, 445, * \rangle[45\%]\}$, where $[\cdot]$ indicates the percentage of flows captured by the corresponding dominant state.

For want of space, in this paper we do not provide a formal treatment of the dominant state analysis. Instead in Fig. 6 we depict the general procedure we use to extract dominant states from a cluster. Let $\{A, B, C\}$ be a re-ordering of the three free dimensions X, Y, Z of the cluster based on their RU values: A is the free dimension with the lowest RU, B the second lowest,

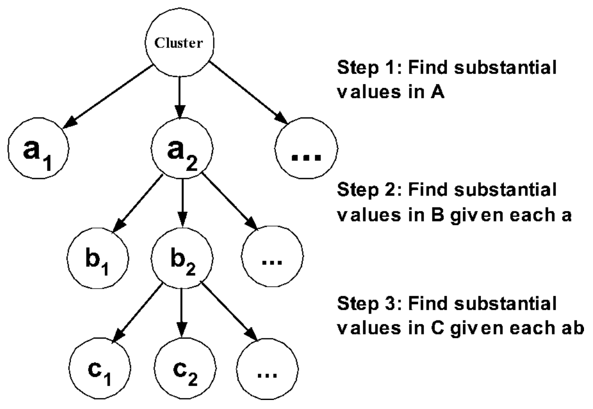
Cluster
a_1
a_2
...
b_1
b_2
...
c_1
c_2
...
Step 1: Find substantial values in A
Step 2: Find substantial values in B given each a
Step 3: Find substantial values in C given each ab

Fig. 6. General procedure for dominant state analysis.

and C the highest; in case of a tie, X always precedes Y or Z, and Y precedes Z. The dominant state analysis procedure starts by finding substantial values in the dimension A (step 1). A specific value a in the dimension A is substantial if the marginal probability $p(a) := \sum_b \sum_c p(a,b,c) \geq \delta$, where δ is a threshold for selecting substantial values. If no such substantial value exists, we stop. Otherwise, we proceed to step 2 and explore the "dependence" between the dimension A and dimension B by computing the conditional (marginal) probability of observing a value b_j in the dimension B given a_i in the dimension $A : p(b_j|a_i) := \sum_c p(a_i,b_j,c)/p(a_i)$. We find those substantial b_j's such that $p(b_j|a_i) \geq \delta$. If no substantial value exists, the procedure stops. Otherwise, we proceed to step 3 compute the conditional probability, $p(c_k|a_i,b_j)$, for each a_i, b_j and find those substantial c_k's, such that $p(c_k|a_i,b_j) \geq \delta$. The dominant state analysis procedure produces a set of dominant states of the following forms: $(*,*,*)$ (i.e., no dominant states), or $a_i \rightarrow (*,*)$ (by step 1), $a_i \rightarrow b_j \rightarrow *$ (by step 2), or $a_i \rightarrow b_j \rightarrow c_k$ (by step 3). The set of dominate states is an approximate summary of the flows in the cluster, and in a sense captures the "most information" of the cluster. In other words, the set of dominant states of a cluster provides a compact representation of the cluster.

We apply the dominant state analysis to the clusters of four feature dimensions extracted on all links with varying δ in [0.1, 0.3]. The results with various δ are very similar, since the data is amenable to compact dominant state models. Table III (ignoring columns 4–7 for the moment, which we will discuss in the next subsection) shows dominant states of `srcIP` clusters extracted from link L_1 over a 1-hour period using $\delta = 0.2$. For each BC, the first row gives the total number of clusters belonging to the BC during the 1-hour period (column 2) and the general or prevailing form of the structural models (column 3) for the clusters. The subsequent rows detail the specific structural models shared by subsets of clusters and their respective numbers. The notations `dstIP`$(\cdot)$, `srcPrt`$(\cdots)$, etc., indicate a specific value and multiple values (e.g., in `dstIP`) that are omitted for clarity, and [>90%] denotes that the structural model captures at least 90% of the flows in the cluster (to avoid too much clutter in the table, this information is only shown for clusters in BC_2). The last column provides brief comments on the likely nature of the flows the clusters contain, which will be analyzed in more depth in Section VI.

The results in the table demonstrate two main points. First, clusters within a BC have *(nearly) identical* forms of structural models; they differ only in specific values they take. For example, BC_2 and BC_{20} consist mostly of hosts engaging in various scanning or worm activities using known exploits, while `srcIP` clusters in BC_6, BC_7 and BC_8 are servers providing well-known services. They further support our assertion that our RU-based behavior classification scheme automatically groups together clusters with similar behavior patterns, despite that the classification is done *oblivious* of specific feature values that flows in the clusters take. Second, the structural model of a cluster presents a compact summary of its constituent flows by revealing the essential information about the cluster (substance feature values and interaction among the free dimensions). It in itself is useful, as it provides *interpretive value* to network operators for understanding the cluster behavior. These observations also hold for clusters extracted from other dimensions and links we studied [10].

B. Exploring Additional Cluster Features

We now investigate whether additional features (beyond the four basic features, `srcIP`, `dstIP`, `srcPrt` and `dstPrt`) can i) provide further affirmation of similarities among clusters within a BC, and in case of wide diversity, ii) be used to distinguish subclasses of behaviors within a BC. Examples of additional features we consider are cluster sizes (defined in total flow, packet and byte counts), average packet/byte count per flow within a cluster and their variability, etc. In the following we illustrate the results of additional feature exploration using the average flow sizes per cluster and their variability.

For each flow f_i, $1 \leq i \leq m$, in a cluster, let PKT_i and BT_i denote the number of packets and bytes respectively in the flow. Compute the average number of packets and bytes for the cluster, $\mu(PKT) = \sum_i PKT_i/m$, $\mu(BT) = \sum_i BT_i/m$. We also measure the flow size variability in packets and bytes using *coefficient of variance*, $CV(PKT) = \sigma(PKT)/\mu(PKT)$ and $CV(BT) = \sigma(BT)/\mu(BT)$, where $\sigma(PKT)$ and $\sigma(BT)$ are the standard deviation of PKT_i and BT_i.

In Table III, columns 4–7, we present the ranges of $\mu(PKT)$, $CV(PKT)$, $\mu(BT)$ and $CV(BT)$ of subsets of clusters with the similar dominant states, using the 1-hour `srcIP` clusters on L_1. Columns 4–7 in the top row of each BC are high-level summaries for clusters within a BC (if it contains more than one cluster): small, medium or large average packet/byte count, and low or high variability. We see that for clusters within BC_6, BC_7, BC_8 and BC_{18}, BC_{19}, the average flow size in packets and bytes are at least 5 packets and 320 bytes, and their variabilities ($CV(PKT)$ and $CV(BT)$) are fairly high. In contrast, clusters in BC_2 and BC_{20} have small average flow size with low variability, suggesting most of the flows contain a singleton packet with a small payload. The same can be said of most of the less popular and rare BCs.

Finally, Fig. 7(a)–(d) show the average cluster sizes[3] in flow, packet and byte counts for all the unique clusters from the dataset L_1 within four different groups of BC's (the reason for the grouping will be clear in the next section): $\{BC_6, BC_7, BC_8\}$, $\{BC_{18}, BC_{19}\}$, $\{BC_2, BC_{20}\}$, and the

[3]We compute the average cluster size for clusters appearing twice or more.

TABLE III
DOMINANT STATES FOR `srcIP` CLUSTERS ON L_1 IN A 1-HOUR PERIOD: $\delta = 0.2$

`srcIP` BC's	No. of Clusters	Structural Models	Range of $\mu(PKT)$	Range of $CV(PKT)$	Range of $\mu(BT)$	Range of $CV(BT)$	Brief Comments
BC_2 $[0,0,2]$	119	srcPrt(·)→dstPrt(·)→dstIP(*)	small	low	small	low	mostly ICMP or scanning traffic
	114	srcPrt(0)→dstPrt(0)→dstIP(*)[>99%]	[1,2]	[0,1.6]	[72,92]	[0,8.9]	ICMP traffic
	1	srcPrt(1026)→dstPrt(137)→dstIP(*)[100%]	1	0	78	0	137: NetBIOS
	1	srcPrt(1153)→dstPrt(1434)→dstIP(*)[>98%]	1	0	404	0	1434: MS SQL
	3	srcPrt(220)→dstPrt(6129)→dstIP(*)[100%]	[1,2]	[0, 1.2]	[40,80]	[0,2.6]	6129: Dameware
BC_6 $[0,2,0]$	16	srcPrt(·)→dstIP(· · ·)→dstPrt(*)	large	high	large	high	server replying to a few hosts
	2	srcPrt(25)→dstIP(· · ·)→dstPrt(*)	[10,15]	[1041,2217]	[120,750]	[36,102]	25: Email
	5	srcPrt(53)→dstIP(· · ·)→dstPrt(*)	[1,5]	[8.6,78]	[160,380]	[111,328]	53: DNS
	7	srcPrt(80)→dstIP(· · ·)→dstPrt(*)	[3,31]	[460,1.2 $*$ 10^4]	[195,1.2 $*$ 10^5]	[16,1612]	80: Web
	2	srcPrt(443)→dstIP(· · ·)→dstPrt(*)	[3,12]	[320,1.5 $*$ 10^4]	[2166,1.1 $*$ 10^5]	[29,872]	443: https
BC_7 $[0,2,1]$	19	srcPrt(·)→dstIP(· · ·)→ dstPrt(*)	large	high	large	high	server replying to many hosts
	2	srcPrt(25)→dstIP→dstPrt(*)	[14,35]	[1129,1381]]	[2498,3167]]	[190,640]	25: Email
	17	srcPrt(80)→dstIP→dstPrt(*)	[4,26]	[210,9146]	[671,1.0 $*$ 10^4]	[29,3210]	80: Web
BC_8 $[0,2,2]$	7	srcPrt(.)→(dstPrt(*),dstIP(*))	large	high	large	high	server replying to large # of hosts
	7	srcPrt(80)→(dstPrt(*),dstIP(*))	[4,27]	[1282,1.1 $*$ 10^4]	[740, 1.5 $*$ 10^4]	[72, 598]	80: Web
BC_{18} $[2,0,0]$	10	dstPrt(·)→(·)dstIP→srcPrt(*)	medium	high	medium	high	host talking to a server on fixed `dstPrt`
	3	dstPrt(53)→dstIP→srcPrt(*)	[2,5]	[32,1.5 $*$ 10^5]	[120,325]	[82,878]	53: DNS
	7	dstPrt(80)→dstIP→srcPrt(*)	[3,18]	[26,6869]	[189,1728]	[87,5086]	80: Web
BC_{19} $[2,0,1]$	6	dstPrt(·)→dstIP(*)→srcPrt(*)	medium	high	medium	high	host talking to multiple hosts on fixed `dstPrt`
	2	dstPrt(53)→dstIP(*)→srcPrt(*)	[2,6]	[28,875]	[116,380]	[112,456]	53: DNS
	3	dstPrt(80)→dstIP(*)→srcPrt(*)	[4,16]	[72.3356]	[220,2145]	[122,2124]	80: Web
	1	dstPrt(7070)→dstIP(*)→srcPrt(*)	3	462	288	261	7070: RealAudio
BC_{20} $[2,0,2]$	58	dstPrt(·)→(srcPrt(*),dstIP(*))	small	low	small	low	host talking to large # hosts on fixed `dstPrt`
	44	dstPrt(135)→(srcPrt(*),dstIP(*))	[1,2]	[0,1.6]	[48,96]	[0,2.7]	135: Microsoft RPC
	1	dstPrt(137)→(srcPrt(*),dstIP(*))	1	0	78	0	137: NETBIOS
	2	dstPrt(139)→(srcPrt(*),dstIP(*))	3	0	144	0	139: NETBIOS
	2	dstPrt(445)→(srcPrt(*),dstIP(*))	[1,3]	[0,2.2]	[48,144]	[0,3.6]	445: Microsoft-DS
	1	dstPrt(593)→(srcPrt(*),dstIP(*))	1	0	48	0	593: http RPC
	2	dstPrt(901)→(srcPrt(*),dstIP(*))	[1,2]	[0,1.6]	[48,96]	[0,3.9]	901: SMPNAMERES
	3	dstPrt(3127)→(srcPrt(*),dstIP(*))	[1,3]	[0,1.8]	[48,144]	[0,2.9]	3127: myDoom worm
	1	dstPrt(6129)→(srcPrt(*),dstIP(*))	1	0	40	0	6129: Dameware
	1	dstPrt(17300)→(srcPrt(*),dstIP(*))	1	0	48	0	17300: unknown
	1	dstPrt(34816)→(srcPrt(*),dstIP(*))	1	0.2	64	0.5	34816: unknown
BC_{24} $[2,2,0]$	1	dstIP(.)→srcPrt(*)→dstPrt(*)	-	-	-	-	two hosts chatting on random ports
	1	dstIP(.)→srcPrt(*)→dstPrt(*)	1	0	889	0	vertical scan

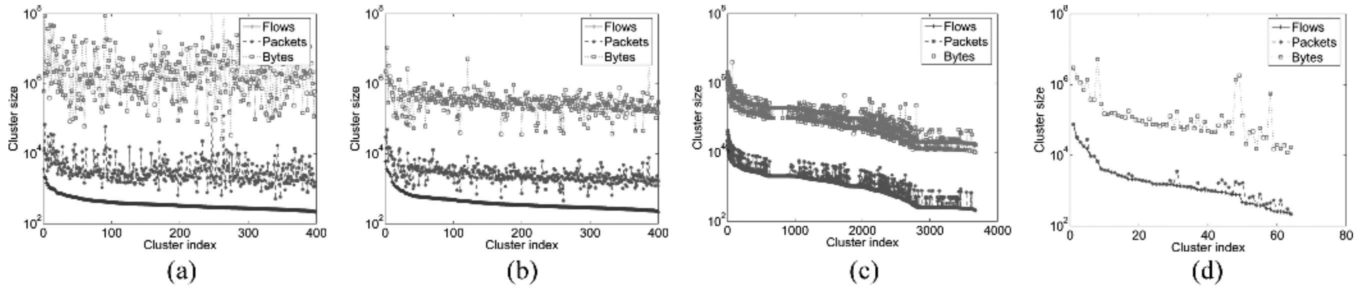

Fig. 7. Average cluster size (in flow, packet and byte count) distributions for clusters within four groups of BC's for srcIP clusters on L_1. Note that in (c) and (d), the lines of flow count and packet count are indistinguishable, since most flows in the clusters contain a singleton packet. (a) BC_6, BC_7, BC_8. (b) BC_{18}, BC_{19}. (c) BC_2, BC_{20}. (d) Other BC's.

fourth group containing the remaining less popular BC's. Clearly, the characteristics of the cluster sizes of the first two BC groups are quite different from those of the second two BC groups. We will touch on these differences further in the next section. To conclude, our results demonstrate that BC's with distinct behaviors (e.g., non-akin BC's) often also manifest dissimilarities in other features. Clusters within a BC may also exhibit some diversity in additional features, but in general the intra-BC differences are much less pronounced than inter-BC differences.

VI. CANONICAL BEHAVIOR PROFILES

We apply our methodology to obtain general profiles of the Internet backbone traffic based on the datasets listed in Table I. We find that a large majority of the (significant) clusters fall into three "canonical" profiles: typical *server/service behavior* (mostly providing well-known services), typical *"heavy-hitter" host behavior* (predominantly associated with well-known services) and typical *scan/exploit behavior* (frequently manifested by hosts infected with known worms). The canonical behavior

TABLE IV
THREE CANONICAL BEHAVIOR PROFILES

Profile	Dimension	BCs	Examples
Servers or Services	`srcIP`	$BC_{6,7,8}$	web, DNS, email
	`dstIP`	$BC_{18,19,20}$	
	`srcPrt`	BC_{23}	aggregate service traffic
	`dstPrt`	BC_{25}	
Heavy-hitter Hosts	`srcIP`	$BC_{18,19}$	NAT boxes
	`dstIP`	$BC_{6,7}$	web proxies, crawlers
Scans or Exploits	`srcIP`	$BC_{2,20}$	scanners, exploits
	`dstIP`	$BC_{2,8}$	scan targets
	`dstPrt`	$BC_{2,5,20,23}$	aggregate exploit traffic

profiles are characterized along the following four key aspects: 1) BCs they belong to and their properties; 2) temporal characteristics (frequency and stability) of individual clusters; 3) dominant states; and 4) additional attributes such as average flow size in terms of packet and byte counts and their variabilities.

A. Server/Service Behavior Profile

As shown in Table IV, a typical server providing a well-known service shows up in either the popular, large and non-volatile `srcIP` $BC_6[0,2,0]$, $BC_7[0,2,1]$ and $BC_8[0,2,2]$, or `dstIP` $BC_{18}[2,0,0]$, $BC_{19}[2,0,1]$ and $BC_{20}[2,0,2]$ (note the symmetry between the `srcIP` and `dstIP` BCs, with the first two labels (`srcPrt` and `dstPrt`) swapped). These BCs represent the behavior patterns of a server communicating with a few, many or a large number of hosts. In terms of their temporal characteristics, the individual clusters associated with servers/well-known services tend to have a relatively high frequency, and almost all of them are stable, re-appearing in the same or akin BCs. The average flow size (in both packet and byte counts) of the clusters shows high variability, namely, each cluster typically consists of flows of different sizes.

An overwhelming majority of the `srcIP` clusters in $BC_{6,7,8}$ are corresponding to Web, DNS or Email servers. They share very similar behavior characteristics, belonging to the same BC's, stable with relatively high frequency, and containing flows with diverse packet/byte counts. Among the remaining clusters, most are associated with http-alternative services (e.g., 8080), https (443), real audio/video servers (7070), IRC servers (6667), and peer-to-peer (P2P) servers (4662). Most interestingly, we find three `srcIP` clusters with service ports 56192, 56193 and 60638. They share similar characteristics with web servers, having a frequency of 12, 9 and 22 respectively, and with diverse flow sizes both in packet and byte counts. These observations suggest that they are likely servers running on unusual high ports. Hence, these cases represent examples of "novel" service behaviors that our profiling methodology is able to uncover.

Looking from the `srcPrt` and `dstPrt` perspectives, the clusters associated with the well-known service ports almost always belong to the same BC's, e.g., either `srcPrt` $BC_{23}[2,1,2]$ or `dstPrt` $BC_{25}[2,2,1]$, representing the aggregate behavior of a (relatively smaller) number of servers communicating with a much larger number of clients on a specific well-known service port.

B. Heavy-Hitter Host Behavior Profile

The second canonical behavior profile is what we call the *heavy-hitter* host profile, which represents hosts (typically clients) that send a large number of flows to a single or a few other hosts (typically servers) in a short period of time (e.g., a 5-minute period). They belong to either the popular and non-volatile `srcIP` $BC_{18}[2,0,0]$ or $BC_{19}[2,0,1]$, or the `dstIP` $BC_6[0,2,0]$ and $BC_7[0,2,1]$. The frequency of individual clusters is varied, with a majority of them having medium frequency, and almost all of them are stable. These heavy-hitter clusters are typically associated with well-known service ports (as revealed by the dominant state analysis), and contain flows with highly diverse packet and byte counts. Many of the heavy-hitter hosts correspond to NAT boxes (many clients behind a NAT box making requests to a few popular web sites, making the NAT box a heavy-hitter), web proxies, cache servers or web crawlers.

For example, we find that 392 and 429 unique `srcIP` clusters from datasets L_1 and L_2 belong to BC_{18} and BC_{19}. Nearly 80% of these heavy-hitters occur in at least 5 time slots, exhibiting consistent behavior over time. The most frequent ports used by these hosts are TCP port 80 (70%), UDP port 53 (15%), TCP port 443 (10%), and TCP port 1080 (3%). However, there are heavy-hitters associated with other rarer ports. In one case, we found one `srcIP` cluster from a large corporation talking to one `dstIP` on TCP port 7070 (RealAudio) generating flows of varied packet and byte counts. It also has a frequency of 11. Deeper inspection reveals this is a legitimate proxy, talking to an Audio server. In another case, we found one `srcIP` cluster talking to many `dstIP` hosts on TCP port 6346 (Gnutella P2P file sharing port), with flows of diverse packet and byte counts. This host is thus likely a heavy file downloader. These results suggest that the profiles for heavy-hitter hosts could be used to identify these unusual heavy-hitters.

C. Scan/Exploit Profile

Behaviors of hosts performing scans or attempting to spread worms or other exploits constitute the third canonical profile. Two telling signs of typical scan/exploit behavior [11] are i) the clusters tend to be highly volatile, appearing and disappearing quickly, and ii) most flows in the clusters contain one or two packets with fixed size, albeit occasionally they may contain three or more packets (e.g., when performing OS fingerprinting or other reconnaissance activities). For example, we observe that most of the flows using TCP protocol in these clusters are failed TCP connections on well-known exploit ports. In addition, most flows using UDP protocol or ICMP protocol have a fixed packet size that matches widely known signature of exploit activities, e.g., UDP packets with 376 bytes to destination port 1434 (Slammer Worm), ICMP packets with 92 bytes (ICMP ping probes). These findings provide additional evidence to confirm that such clusters are likely associated with scanning or exploit activities.

A disproportionately large majority of extracted clusters fall into this category, many of which are among the top in terms of flow counts (but in general not in byte counts, cf. Fig. 7). These hosts manifest distinct behavior that is clearly separable from the server/service or heavy-hitter host profiles: the `srcIP` clusters (a large majority) belong to $BC_2[0,0,2]$ and $BC_{20}[2,0,2]$, corresponding to hosts performing scan or spreading exploits to random `dstIP` hosts on a fixed `dstPrt` using either fixed or random `srcPrt`'s; the `dstIP` clusters (a smaller number) belong to $BC_2[0,0,2]$ and $BC_8[0,2,2]$, reflecting hosts (victims of a large number of scanners or attacks) responding to probes on a targeted `srcPrt`.

In addition to those `dstPrt`'s that are known to have exploits, we also find several (`srcIP`) clusters that manifest typical scan/exploit behavior, but are associated with `dstPrt`'s that *we do not know* to have known exploits. For example, we find that in one time slot a `srcIP` cluster is probing a large number of destinations on UDP port 12827, with a single UDP packet. This host could simply engage in some harmless scanning on UDP port 12827, but it could also be a new form of RATs (remote access trojans) or even a precursor of something more malicious. Further inspection is clearly needed. Nonetheless it illustrates that our profiling technique is capable of automatically picking out clusters that fit the scan/exploit behavior profile but with unknown feature values. This will enable network operators/security analysts to examine novel, hitherto unknown, or "zero-day" exploits.

D. Deviant or Rare Behaviors

We have demonstrated how we are able to identify novel or anomalous behaviors that fit the canonical profiles but contain unknown feature values (as revealed by the dominant state analysis). We now illustrate how rare behaviors or deviant behaviors are also indicators of anomalies, and thus worthy of deeper inspection. In the following, we present a number of case studies, each of which is selected to highlight a certain type of anomalous behavior. Our goal here is not to exhaustively enumerate all possible deviant behavioral patterns, but to demonstrate that building a comprehensive traffic profile can lead to the identification of such patterns.

Clusters in Rare Behavior Classes: The clusters in the rare behavior classes by definition represent atypical behavioral patterns. For example, we find three `dstPrt` clusters (TCP ports 6667, 113 and 8083) suddenly appear in the rare `dstPrt` $BC_{15}[1,2,0]$ in several different time slots, and quickly vanish within one or two time slots. Close examination reveals that more than 94% of the flows in the clusters are destined to a single `dstIP` from random `srcIP`'s. The flows to the dstIP have the same packet and byte counts. This evidence suggests that these `dstIP`'s are likely experiencing a DDoS attack.

VII. Related Work

Most of the prior work has analyzed specific aspects of traffic or applied metrics that are deemed interesting *a priori* to identify significant network events of interest. For example, [12], [13] focus on efficient techniques for identifying "heavy-hitters" in one or several dimensions, and [14], [15] focus on identifying port scans. In [16], Zhang *et al.* present streaming algorithms for detecting multidimensional hierarchical heavy-hitters. Mahoney *et al.* introduce a two-stage anomaly detection system for identifying suspicious traffic for well-known applications, such as FTP, HTTP and SMTP in [17]. In contrast to both of these works, our goal in this work is to build behavior profiles for all significant hosts or services, not specific traffic patterns or applications.

[18] studies the behavior of flash crowds, while [19]–[21] focus on analyzing worm and other exploit activities on the Internet. Research in [22], [23] applies signal processing and statistical inference techniques for identifying traffic anomalies, mostly from the perspective of link-level traffic aggregates. Signature-based intrusion detection systems look for well-known signatures or patterns in network traffic, while several behavior-based anomaly detection systems (see, e.g., [24], [25] and references therein) have been developed using data mining techniques. In [26], information-theoretic measures are proposed for evaluating anomaly detection schemes. All of these works are interested in one or more specific behaviors, while ours focuses on understanding common behaviors, including normal or anomalous behaviors.

In [27], Hao *et al.* consider the problem of detecting hidden traffic patterns by examining packet streams. The hidden traffic detection algorithm proposed in [27] is efficient for detecting high-volume flows without knowing flow dimensions a priori. However, this approach requires a pre-defined threshold, which is often hard to predict in backbone links.

Closer to our work, [4] focuses on resource consumption in network traffic, and develops a clustering algorithm that automatically discovers significant traffic patterns along one or multiple dimensions using fixed volume thresholds. The studies in [28], [29] focus on communication patterns or profiles of applications instead of broader network traffic. Concurrent with our work, [30], [31] are most similar in spirit, and in a sense are complementary, to ours. In [30], the authors study the "host behaviors" (communication patterns) at three levels, with the objective to classify traffic flows using packet header information only. As an extension to their early work [22], [23], the authors in [31] also use entropy to characterize traffic feature distributions, with emphasis on detecting *network-wide* traffic anomalies at PoP-level OD (origin-destination) flows: the PCA-based subspace method is used to separate "anomalies" from "normal" traffic. In contrast, our objective is to build behavior profiles at host and service levels using traffic communication patterns without any presumption on what is normal or anomalous.

VIII. Conclusion

Extracting significant events from vast masses of Internet traffic has assumed critical importance in light of recent cyber attacks and the emergence of new and disruptive applications. In this paper, we have used data-mining and entropy-based techniques to automatically discover significant behavior patterns from link-level traffic data, and to provide plausible interpretations for the observed behaviors. We have demonstrated the applicability of our profiling approach to the problem of detecting unwanted traffic and anomalies. We also have investigated possible countermeasure strategies that a backbone ISP may pursue for reducing unwanted exploit traffic based on their characteristics [11]. Our results demonstrated that blocking the most offending sources is reasonably cost-effective. In [32], through extensive performance benchmarking of CPU and memory costs, we demonstrated the feasibility of implementing and utilizing a real-time behavior profiling system for high-speed Internet links. We are currently studying the implications and potential benefits of extending our profiling approach beyond flow-level header information to application-level payload carried in IP packets.

REFERENCES

[1] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Chicago, NY: Univ. Illinois Press, 1949.

[2] T. Cover and J. Thomas, *Elements of Information Theory*, ser. Wiley Series in Telecommunications. New York: Wiley, 1991.

[3] K. Claffy, H.-W. Braun, and G. Polyzos, "A parameterizable methodology for internet traffic flow profiling," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 8, pp. 1481–1494, Oct. 1995.

[4] C. Estan, S. Savage, and G. Varghese, "Automatically inferring patterns of resource consumption in network traffic," in *Proc. ACM SIGCOMM*, Sep. 2003, pp. 137–148.

[5] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Profiling internet backbone traffic: Behavior models and applications," in *Proc. ACM SIGCOMM*, Aug. 2005, pp. 169–180.

[6] K. Krippendorff, *Information Theory: Structural Models for Qualitative Data*. Thousand Oaks, CA: Sage, 1986.

[7] R. Cavallo and G. Klir, "Reconstructability analysis of multi-dimensional relations: A theoretical basis for computer-aided determination of acceptable systems models," *Int. J. General Syst.*, vol. 5, pp. 143–171, 1979.

[8] M. Zwick, "An overview of reconstructability analysis," *Int. J. Syst. Cybern.*, vol. 33, pp. 877–905, 2004.

[9] M. Jordan, "Graphical models," *Statist. Sci., Special Issue Bayesian Statistics*, vol. 19, pp. 140–155, 2004.

[10] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Profiling Internet backbone traffic: Behavior models and applications," Sprint ATL Res. Rep. RR05-ATL-020777, Feb. 2005.

[11] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Reducing unwanted traffic in a backbone network," in *Proc. Steps Reducing Unwanted Traffic Internet Workshop (SRUTI)*, Jul. 2005, pp. 9–15.

[12] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection: Methods, evaluation, and applications," in *Proc. ACM/USENIX IMC*, 2003, pp. 234–247.

[13] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Finding hierarchical heavy hitters in data streams," in *Proc. VLDB*, 2003, pp. 464–474.

[14] S. Staniford, J. Hoagland, and J. McAlerney, "Practical automated detection of stealthy portscans," *J. Comput. Security*, vol. 10, pp. 105–136, 2002.

[15] J. Jung, V. Paxson, A. Berger, and H. Balakrishna, "Fast portscan detection using sequential hypothesis testing," in *Proc. IEEE Symp. Security Privacy*, 2004, pp. 211–225.

[16] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund, "Online identification of hierarchical heavy hitters: Algorithms, evaluation, and applications," in *Proc. Internet Meas. Conf.*, 2004, pp. 101–114.

[17] M. Mahoney, "Network traffic anomaly detection based on packet bytes," in *Proc. ACM Symp. Appl. Comput.*, Mar. 2003, pp. 346–350.

[18] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites," in *Proc. Int. WWW Conf.*, 2002, pp. 293–304.

[19] N. Weaver, V. Paxon, S. Staniford, and R. Cunningham, "A taxonomy of computer worms," in *Proc. CCS Workshop Rapid Malcode (WORM)*, 2003, pp. 11–18.

[20] V. Yegneswaran, P. Barford, and J. Ullrich, "Internet intrusions: Global characteristics and prevalence," in *Proc. ACM SIGMETRICS*, 2003, pp. 138–147.

[21] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson, "Characteristics of internet background radiation," in *Proc. ACM SIGCOMM IMC*, 2004, pp. 27–40.

[22] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proc. ACM SIGCOMM*, 2004, pp. 219–230.

[23] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *Proc. IMC*, 2004, pp. 201–206.

[24] MINDS, Minnesota Intrusion Detection System. [Online]. Available: http://www.cs.umn.edu/research/minds/

[25] A. Lazarevic, L. Ertoz, A. Ozgur, J. Srivastava, and V. Kumar, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proc. SIAM Conf. Data Mining*, 2003, pp. 25–36.

[26] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," in *Proc. IEEE Symp. Security Privacy*, 2001, pp. 130–143.

[27] F. Hao, M. Kodialam, and T. Lakshman, "Real-time detection of hidden traffic patterns," in *Proc. ICNP*, Oct. 2004, pp. 340–349.

[28] F. Hernandez-Campos, A. B. Nobel, F. D. Smith, and K. Jeffay, "Statistical clustering of internet communication patterns," in *Proc. Symp. Interface Computing Sci. Statistics*, 2003, p. 134.

[29] S. J. Stolfo, S. Hershkop, K. Wang, O. Nimeskern, and C. Hu, "Behavior profiling of email," in *Proc. NSF/NIJ Symp. Intell. Security Informatics*, 2003, pp. 74–90.

[30] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel traffic classification in the dark," in *Proc. ACM SIGCOMM*, 2005, pp. 229–240.

[31] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *Proc. ACM SIGCOMM*, Aug. 2005, pp. 217–228.

[32] K. Xu, F. Wang, S. Bhattacharyya, and Z.-L. Zhang, "A real-time network traffic profiling system," in *Proc. Int. Conf. Dependable Syste. Netw.*, June 2007, pp. 595–605.

Kuai Xu received the B.S. and M.S. degrees in computer science from Peking University, Beijing, China, in 1998 and 2001, respectively, and the Ph.D. degree in computer science from the University of Minnesota, Minneapolis, in 2006.

He joined network system group of Yahoo! Inc., Sunnyvale, CA, in 2006. His current research lies in the modeling and analysis of network traffic and end-to-end performance in distributed content networks.

Zhi-Li Zhang (M'97) received the B.S. degree from Nanjing University, Nanjing, China, and the M.S. and Ph.D. degrees from the University of Massachusetts, Amherst, all in computer science.

In 1997 he joined the Computer Science and Engineering faculty at the University of Minnesota, Minneapolis, where he is currently the Qwest Chair Professor in Telecommunications. He has held visiting positions at Sprint Advanced Technology Labs, IBM T.J. Watson Research Center, Fujitsu Labs of America, Microsoft Research China, and INRIA, Sophia-Antipolis, France.

Supratik Bhattacharyya received the M.S. and Ph.D. degrees in computer science from the University of Massachusetts, Amherst.

He is currently with SnapTell Inc, Palo Alto, CA. He was a Distinguished Member of Technical Staff at Sprint Advanced Technology Laboratories in Burlingame CA. His work at Sprint has covered a number of aspects of core IP networks such as performance monitoring, routing, traffic engineering and fault tolerance. His current interests are in mobile communication and services and in mining network traffic data.