Sifting through Network Data to Cull Activity Patterns with HEAPs

Esam Sharafuddin, Yu Jin, Nan Jiang, Zhi-Li Zhang
Department of Computer Science and Engineering
University of Minnesota
{shara,yjin,njiang,zhzhang}@cs.umn.edu

***Abstract*—Today's large campus and enterprise networks are characterized by their complexity, i.e. containing thousands of hosts, and diversity, i.e. with various applications and usage patterns. To effectively manage and secure such networks, network operators and system administrators are faced with the challenge of characterizing, profiling and tracking activity patterns passing through their networks. Because of the large number of IP addresses and the prevalence of dynamic IP addresses, profiling and tracking individual hosts may not be effective nor scalable. In this paper, we develop a hierarchical extraction of activity patterns (HEAPs), which is a method for characterizing and profiling activity patterns within subnets. By representing activities within a subnet in a host-port association matrix (HPAM) and applying pLSA, we obtain co-clusters that capture the significant and dominant activity patterns of the subnet. Using these co-clusters, we utilize hierarchical clustering to cluster activity patterns to assist network operators and security analysts gain a "big-picture" view of the network activity-patterns. We also develop a novel method to track and quantify changes in activity patterns within subnets over time and demonstrate how to utilize this method to identify major changes and anomalies within the network.**

I. Introduction

As we increasingly depend on computer networks for communication, information access and storage, entertainment and other activities, managing and securing such networks are critical. However, with new applications and services constantly emerging and becoming more complex, knowing what is going on a network, e.g., what applications and services are running and on what machines, is no longer a simple and easy task. This is particularly true in a large campus or enterprise network with a large number of heterogeneous hosts and devices (e.g., various servers, office desktops, laptops, lab machines, printers, as well as wireless access points, routers and switches) that serve the needs of a diverse user population.

To help figure out what is going on a large campus or enterprise network, in this paper we study the problem of how to automatically extract significant "activity patterns" from passively collected network traffic data which can then be used to identify applications or "clusters" of related applications that are often used together (e.g., as part of a service) as well as groups of hosts on which these applications/services are frequently operating. More specifically, we use the term *activity pattern* to refer to *a collection of transport layer (TCP/UDP) ports and end hosts that are strongly associated*. In other words, given a set of ports and a group of hosts as represented by an activity pattern, not only these ports are frequently used by (applications running on) these hosts, but also they are used in similar manners (in Section II, we will provide a formal *probabilistic* definition using a machine learning model). Thus, an activity pattern *simultaneously* captures the *set of ports* likely used by some (cluster of) applications or services *and* the *group of hosts* they frequently operate on.

As most applications use either fixed, reserved ports or can be mapped to a set of frequently used ports with high probability, using the activity patterns, we can then identify the likely applications/services they represent as well as the groups of hosts they are strongly associated with. In those cases where the ports are unknown, or cannot be reliably mapped, or the hosts with which are strongly associated are unexpected (e.g., source TCP port 25 associated with a client desktop), the activity patterns also provide useful information or alerts to guide network operators for further follow-on analysis (e.g., via signature-based traffic classification, deep packet inspection or other sources of data). Hence, activity patterns extracted from network data can indeed reveal what is running on a network, and on what hosts. We will show later in this paper how activity patterns not only provide a big picture of what is going on a large campus network, but can also be used to profile and track individual subnets for network management and security purposes, for example, to discover emergence of new disruptive applications or detect security threats that affect either a specific subnet or a big portion of the whole network.

The remainder of the paper is therefore centered on the development of an effective and scalable methodology for automatically extracting activity patterns from massive network data. We put forth a novel methodology called *HEAPs—Hierarchical Extraction of Activity Patterns*, which employs a *local-global* two-stage process for both scalability and effectiveness. In the first stage, we employ the *probabilistic Latent Semantic Analysis* (pLSA) method to extract *locally significant* activity patterns using network data from *individual network blocks (subnets)*. Using the (significant) activity patterns extracted from individual subnets, in the second stage, we perform a *global classification* of activity patterns by analyzing the similarities of locally extracted activity patterns and grouping them accordingly using a hierarchical clustering

The work is supported in part by the National Science Foundation grants CNS-0626808, CNS-0626812, CNS-0905037 and the DTRA grant HDTRA1-09-1-0050.

algorithm. Together, these two stages produce an effective and scalable process for extracting and classifying activity patterns that either are globally (i.e., network-wide) prevalent or have local (i.e., subnet-wise) significance. In Section II, we describe the network data used in our study, motivate why probabilistic Latent Semantic Analysis (pLSA) is a good model for extracting significant activity patterns from network data based on host and port associations, and discuss the issues involved in applying pLSA. We then provide an overview of the proposed HEAPs, and conclude with a brief discussion of alternative approaches and related works. In Section III, we describe the detailed steps in applying pLSA to extract significant activity patterns locally using network data from individual subnets (address blocks)—the first (local) stage of HEAPs. Using real network data, we illustrate the (locally extracted) activity patterns via examples, show how to infer the applications associated with these activity patterns, and discuss the effect of the size of subnets on the efficacy and scalability of pLSA. In Section IV, we employ the *Jensen-Shannon* distance metric to analyze and compare the activity patterns extracted from individual subnets, and apply a hierarchical clustering algorithm to classify them across subnets so as to produce a *network-wide* picture of what is going on in the network—the second (global) stage of HEAPs. Using the extracted activity patterns and their classification, we demonstrate how HEAPs can be used to profile and track what applications are running on subnets. Using examples, we show that subnets of a large network with diverse user populations often exhibit diverse behaviors: while some applications are common to all subnets, others may show up only in some subnets but not others, indicative of the functions of these subnets and the user behaviors on them. Such diverse behaviors of different subnets are also evident when we study the temporal properties of block activity patterns in Section V, which also provide hints on the composition of hosts in each block (we use block and subnet interchangeably). To further illustrate the utility of HEAPs, in Section V, we artificially inject "suspicious" or "attack" traffic into the real network data and show how they can be detected by tracking the change of block activities and extracting *anomalous* activity patterns on individual subnets. The paper is concluded in Section VI.

II. Motivation and Overview

We start with a brief description of the datasets. We then motivate the main problem addressed in the paper, present the pLSA model for activity extraction, and provide an overview of HEAPs. We conclude by briefly discussing alternative approaches and related work.

A. Datasets

The datasets used in our study come from a large university campus network. The campus network contains 3 class B (i.e., /16) address blocks, and is composed of a wide range of disparate *subnets* (e.g., various college/departmental subnets, lab subnets, supercomputer centers, residential subnets, public wireless access subnets, etc.), serving the needs of a large diverse user population (faculty, staff, students and other users). The datasets include (unsampled) bidirectional Cisco NetFlow records collected at a border router of the campus network over one month period, corresponding to traffic between inside (campus) IP addresses (or hosts) and outside IP addresses (hosts). For this study, we use the *outgoing* TCP, UDP and ICMP traffic, which account for more than 99% of all the traffic from the campus network to the outside Internet hosts. The outgoing flows represent traffic activities either initiated by inside hosts, or in response to outside service requests; thus they are indicative of true activities of the inside hosts. In contrast, incoming traffic may contain a significant amount of "noises" (such as various scanning, backscatter and other activities) generated by the outside hosts toward our campus network, many of which do not even pass the campus network border firewall. In the following, when referring to specific source or destination port number, e.g., source or destination port 80, we will also use the shorthand *srcPort 80* or *dstPort 80*. We will also treat ICMP packets *as if they were transport layer flows*, using *srcPort xx* and *dstPort xx* to represent the corresponding request/response codes contained in the ICMP packets (as done in Cisco NetFlow records).

B. Host-Port Associations and Activity Pattern Extraction

Let H denote the set of all "active" IP addresses (i.e., *inside hosts*) that are observed to generate some traffic activities over a given time window T (throughout the paper, we will set T to one day, from 0th hour to 24th hour, using the timestamps in the NetFlow records). For each host $h \in H$, we can construct a *port distribution vector* (PDV) using the observed traffic from h over the time window T. The PDV consists of two parts, *src* PDV and *dst* PDV, each of which records a probability of a given *src* (resp. *dst*) port is observed over T. The probability is computed as the number of flows containing the said *src* (resp. *dst*) port over the total number of flows. Using these PDVs of inside hosts, we would like to a) identify "classes" or "clusters" of related applications that are frequently used in the network, and b) groups of hosts on which these applications are often run, with the goal to glean a big picture of what is going on in the network. Due to the diversity of the large campus network, we are not only interested in the significant activities that are prevalent across a large portion of the entire network, but also those that are of significance to individual subnets. This is partly motivated by the often disparate management and security policies that are applicable to different subnets. For instance, file sharing applications are not allowed (unless for the purpose of research) in departmental subnets, whereas they are allowed in residential subnets. In addition, due to their differing missions and user needs, certain subnets need to be better managed and protected than others.

As mentioned earlier in the introduction, there are strong intrinsic associations between Internet applications and the transport layer port numbers observed in the network traffic. For instance, many popular applications and services use so-called *reserved ports* as approved by IANA, e.g., TCP port 80 for web applications; TCP port 25 and other related ports (e.g., 465, 587) for email applications; UDP/TCP port 53 for DNS; TCP ports 6665-6669, or 6679 for the Internet

Relay Chat (IRC) service; TCP port 1214 for Kazaa; and TCP/USP port 6346 for Gnutella (the last two are file sharing applications), and so forth. Many other applications, while may not have reserved ports, frequently use (either by default or common practice) certain ports or port ranges: for example, Yahoo Messenger always uses TCP port 5050; Windows Live Messenger services always use ports (TCP/UDP) within the range 6891-6901, BitTorrent often uses some part of the TCP/UDP port range 6881-6900; TCP 23399 is used (as default) by Skype; and so on. Such strong association between applications and TCP/UDP ports is fundamentally an artifact of the design of transport layer protocols and socket APIs, where at least one endpoint of a communication over the network must listen on a pre-specified port in order to receive data, where the application is client-server or peer-to-peer. Although different ports may be used by different instances of the application, once it has been configured for the application instance running on a host, more often than not it tends to stay unchanged. While it is possible to compile an exhaustive list to find ports reserved or frequently used by various applications (such a list is in fact used for mapping ports to plausible applications for the purpose of validating and interpreting the results we have obtained), such a *manual* approach is rather tedious; nor is it generally scalable or robust. For example, simply mapping certain "reserved" ports to their "officially assigned" applications may not always be a good idea, as some other applications may re-use or "mis-use" these ports for other purposes. Further, the associations between applications and ports may be *site-dependent*, especially in the case of applications/services which allow users or system administrators to configure the ports used.

Rather than using ports to directly identify applications, we instead focus on the significant *host-port* association patterns that can be mined from the network data, and employ only a list of known port-application mappings for the purposes of validation and interpretation of extracted activity patterns. The basic idea behind our approach is the following. Suppose we have a collection of hosts (say, most are client machines, but some are servers) as well as a set of applications/services. Each application is operated on a subset of hosts, and each host may run one or more applications. Hence, in general, the groups of hosts running each application may intersect. For simplicity, let us assume that the applications are "uncorrelated" in the sense that how users (of the hosts) use one application do not in general hinge on another application, e.g., users accessing the web do not (with high probability) lead them to run a file sharing application, or use Skype; and moreover, for a group of hosts frequently running the same application, it is used in a somewhat similar manner. Under these assumptions, when observing the frequencies of (*src/dst*) ports in the traffic generated by the individual hosts in the collection, intuitively, we would expect that the ports associated with an application would appear more frequently among the group of hosts on which it is operated than those on which it is not operated. Further, although hosts may operate multiple applications, we would expect the frequencies of the ports associated with one application to be more *correlated* on the hosts it is operated than those it is not. This illustrative example suggests that we may be able to extract significant activity patterns by looking for the *strong host-port association patterns* in the port distribution vectors (PDVs) of the hosts. In the machine learning jargon, extracting significant activity patterns based on the strong host-port associations in the network data is essentially a *co-clustering* problem. In the next section, we adopt the *probabilistic Latent Semantic Analysis* (pLSA) technique developed for document classification for this problem, and provide an overview of the pLSA-based HEAPs method.

C. pLSA and Overview of HEAPs

The probabilistic Latent Semantic Analysis (pLSA) [1] technique is a *generative* machine learning model originally developed for document classification. Here, we first provide a formal description of the model, and then describe how we adopt pLSA to develop *HEAPs* for extracting and classifying activity patterns using a novel *local-global two-step process*.

As an example of the *generative aspect model*, pLSA posits a *latent-variable* model for clustering/classifying and "explaining" observed co-occurrence (or "dyadic") data. It assumes that the observed data (e.g., documents, or in our case traffic generated by hosts) come from K *latent* classes or "aspects" (e.g., document classes as defined by subject topics, or in our case classes or clusters of related applications) that cannot be directly observed and thus must be inferred. What we have is an observed co-occurrence data matrix (e.g., documents and frequencies of words contained in each document, or in our case hosts and the frequencies of ports observed), the problem is then to cluster and classify each data to its latent class (e.g., classify each document based on the topic, or hosts based on the application it runs). Note that the clustering/classification here is in general *soft*, i.e., data (documents, hosts) can be classified into multiple classes, each with a given probability. For clarity, we will cast the pLSA model in terms of extracting the activity patterns using the host-port association data.

Given $H = \{h_1, \ldots, h_m\}$, a collection of hosts, and $P = \{p_1, \ldots, p_n\}$, the set of (*src/dst*) associated ports, let the $m \times n$ matrix A denote the *host-port association matrix* (HPAM), where each entry, $A(h, p)$, represents the (joint) probability (*src* or *dst*) port p is observed to be associated with h in the overall traffic. (Hence each row of A corresponds to (a normalized version of) the PDV of h, see Section III for a formal definition of HPAMs used in our study.) As earlier, we assume that there are K latent (application or "activity pattern") classes, $C = \{c_1, \ldots, c_K\}$. Given each (h, p) pair, pLSA models the probability of the observed co-occurrence, $A(h, p)$, as a mixture of *conditionally independent* multinomial distributions. In other words, a joint probability model over $H \times P$ is defined by the mixture, where

$$A(h,p) := Pr(h,p) = Pr(p|h)P(h) = \sum_{c \in C} Pr(p|c)Pr(c|h)Pr(h), \tag{1}$$

where we have used the relation

$$Pr(p|h) = \sum_{c \in C} Pr(p|c)Pr(c|h), \tag{2}$$

with the assumption that h and p are independent *conditioned on* c, and thus $Pr(p|c,h) = Pr(p|c)$, $\forall c \in C$.

Given this generative aspect model for $A = [Pr(p,h)]$, the problem is then to infer the probabilities $Pr(h|c)$ and $Pr(p|c)$, or equivalently, $Pr(h,p|c)$ $\forall c \in C, h \in H, p \in P$. From these probabilities, we can extract *significant activity patterns*, i.e., the collection of associated hosts and ports such that for some c, $Pr(h,p|c)$ is significant, e.g., exceeds some threshold. In pSLA, these probabilities are estimated using the *Expectation Maximization* (EM) algorithm. As part of the E-step and M-step, the following formulae are used (see [2] for details), where $F(h,p)$ denotes the numbers of flows host h sends using port p:

$$Pr(p|c) \propto \sum_{h \in H} F(h,p) Pr(c|h,p) \tag{3}$$

which is the posterior probability of port p belonging to (the latent activity pattern) c, and

$$Pr(h|c) \propto \sum_{p \in P} F(h,p) Pr(c|h,p) \tag{4}$$

which is the posterior probability of host h belonging to (the latent activity pattern) c. Hence for each $c \in C$,

$$Pr(c) \propto \sum_{h \in H} \sum_{p \in P} F(h,p) Pr(c|h,p). \tag{5}$$

In adopting the pLSA model to extract activity patterns in massive network data, we face several technical challenges. First, because the scale of the data (both in terms of numbers of hosts and ports), pLSA may not be able to handle the large data matrix, due to the high computation costs (of the iterative EM algorithm), especially when the number of latent classes K is large. Second, because of the diversity of applications on a large network, determining the appropriate K can be a difficult issue. Third and perhaps more importantly, extracting *significant* activity patterns using the global (network-wide) data matrix using pLSA may obscure those that are specific to and of significance to individual subnets. This, for example, may happen when a relatively small K is used. To address these issues, we propose *HEAPs—Hierarchical Extraction of Activity Patterns*, a novel *local-global two-stage* process that is not only scalable but also effective in extracting both globally prevalent as well as locally significant activity patterns. In the first stage of HEAPs, we apply pLSA locally to host-port association data derived from network traffic of *individual subnets* to extract significant activity patterns. Using these activity patterns extracted from individual subnets, in the second stage of HEAPs, we use the Jensen-Shannon distance metric to compare activity patterns extracted from different subnets based on their corresponding port distributions (i.e., using $\{Pr(p|c), p \in P\}$ for each (latent) activity pattern c), and then classify them accordingly using a hierarchical clustering algorithm. The second stage therefore enables us to group and identify activity patterns that are *globally* prevalent, while at the same time discovering those that are of significance only to one or a few subnets. We will present the details of these two stages in Section III and Section IV.

D. Related Work

As formulated in this paper, extracting significant activity patterns based on host-port associations is essentially a *co-clustering* problem. In the data mining/machine learning communities, a number of co-clustering techniques have been developed. For example, a classic method is to use SVD (singular value decomposition) or PCA (principal component analysis), as is used in the standard Latent Semantic Analysis (LSA) [3]. While SVD is known to be optimal in minimizing the L_2 norm (i.e., the variance), the main issue with SVD is that the decomposition results can sometimes be hard to interpret, especially when L_2 or variance minimization may not be the right model for how the data is generated. Outliers may also affect the results of SVD, for which a number of *robust* SVD/PCA methods have been developed (see, e.g., [4]). Spectral co-clustering of bipartite graphs [5] essentially applies SVD to bipartite graphs. Another approach for co-clustering employs non-matrix factorization techniques (see, e.g., [6]–[8]) to decompose and cluster co-occurrence data matrices. The main advantage of this approach is that the results are in general easier to interpret (as decomposed matrices are still non-negative, as in the original data matrix). Most non-matrix factorization techniques are computationally expensive, and cannot handle large data matrices. As generative models, pLSA (and the closely related model, LDA [9]) not only produces "non-negative" decomposition results that are more amenable to interpretation, it also supplies us with a specific model (the latent variable mixture model) for explaining how the observed data may be generated. As a plausible generative model for observed host-port association data, we therefore adopt pLSA for our problem.

Within the context of relevant work to this paper, one may apply *traffic classification* directly to identify applications, and then discover hosts associated with these applications. Most traffic classification schemes (see, e.g., [10]–[13]), however, require payload (or at least application layer protocol fields) for signature-based application identification, or detailed traffic statistics for machine-learning based traffic classification. In contrast, our work utilizes much coarser host-port association data (over a generally much longer period of time), and thus is far less inexpensive and more scalable. Our approach and the traffic classification approach are not mutually exclusive, and can be in fact viewed as complementary: our approach can be applied to identify activity patterns with strong host-port associations, and if needed, the traffic classification approach can then be applied to these activity patterns for further in-depth analysis, e.g., to validate and map ports to (expected) application classes, or to discover unknown applications. The combination of the two approaches is especially useful in detecting anomalous and suspicious activities (e.g., attacks) on the network. More closely related to our work, the authors in [14] apply SVD to block-level host-port network matrix for the purpose of profiling and tracking significant behaviors of each network block. In [15], the non-negative matrix factorization approach is used to decompose traffic activity graphs (TAGs) formed by interactions between inside/outside hosts of a campus network, and mine significant interaction

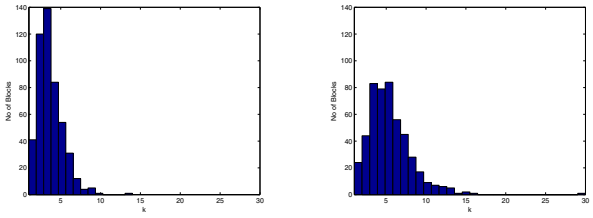
patterns. While both methods can be used in our study, for the reasons discussed earlier we adopt pLSA for its model-based data interpretations. We note that it is possible to incorporate [15] in our work to model 3-way interactions among inside, outside hosts and ports. This will be part of the future work.

III. Extracting Activity Patterns from Subnets

In this section, we describe how to apply the pLSA model to extract significant activity patterns using network data from individual subnets. We present some examples to illustrate the extracted activity patterns, and discuss the impact of address block (subnet) sizes on the efficacy of pLSA. We also show how we utilize the characteristics of activity patterns to infer the underlying applications of a given subnet.

A. Applying pLSA to Subnet HPAMs

As the common practice, each subnet of a large campus network (as in ours) is typically allocated a contiguous block of IP addresses of size 2^b (i.e., a /b address block), where typically we have $16 < b \leq 24$. For simplicity, in this paper we divide the address space (consisting of 3 /16 address blocks) of the campus network into *subnets of block size* /b for some b. For most cases, we will use $b = 24$ unless otherwise stated. In Section III-D, we will discuss the impact of subnet block size b on the efficacy of pLSA.



(a) $\epsilon = 10\%$ (b) $\epsilon = 5\%$

Fig. 1. No of singular values for different values of ϵ

Given a subnet S of a block size b, we have a total of up to $m = 2^b$ hosts. We will use H to denote the set of all hosts on the subnet S. For each $h \in H$, let f_h be the total number of flows generated by h in a time window T (e.g., a day). For each *src* (resp. *dst*) port p, let f_{hp} denote the total number of flows generated by h that contain p as the *src* (resp. *dst*) port. As there are potentially 2^{16} (*src*/*dst*) ports most of which are either not used or infrequently used, without loss of generality, we "compress" the total number of (*src*/*dst*) ports into a list of 4000 most frequently observed (or most significant) ports, with 2000 *src* and 2000 dst ports as follows: Using the flow records from the entire network, for each *src* port p, we compute the frequency of the port (i.e., the ratio of flows containing the *src* p over the total number of flows), and apply the entropy-based significant value (cluster) extraction method developed in [16] to identify the most significant (or frequent) *src* ports. Suppose that this process yields a list of N_0 top *src* ports. Given this list of top N_0 *src* ports, we apply the same method using the flow records of each subnet (i.e., each /b address block) to extract the *subnet-specific* most significant/frequent ports and check to see whether these ports are among the list of top N_0 ports extracted earlier and if they are not, we add those ports to the list. This way we obtain a list of most frequent or significant *src* ports that are either globally frequent or locally significant. Using the same method, we obtain a similar list of most frequent or significant *dst* ports. With some minor adjustment for symmetry, etc., we obtain a final list of 1999 *src* and 1999 *dst* ports. Finally, we introduce one "fictitious" *src* port, denoted as *aoSrcPort*, which represents all other *src* ports; likewise, the "fictitious" *dst* port, denoted as *aoDstPort*, represents all other *dst* ports. We use P to denote this set of 2000 *src* and 2000 *dst* ports, where $n = |P| = 4000$.

Given the set of hosts H and the set of ports P, the *host-port association matrix* (HPAM) for a subnet S is defined as:

$$A_S := [s_h \cdot f_{hp}/f_h]_{m \times n} \tag{6}$$

where $s_h = f_h/f_H$, and $f_H = \sum_{h' \in H} f_{h'}$. Namely, f_H is the total number of flows generated by all hosts on the subnet S, and s_h thus represents the fraction of flows generated by host h. Intuitively, s_h represents the probability we see host h active[1].

When applying pLSA to the host-port association matrix A defined above, a key issue is to determine an appropriate K, the number of latent classes (i.e., significant activity patterns). For determining K, we employ a standard trick. We first apply SVD (singular value decomposition) to A to obtain an initial estimate K'. Using this initial estimate K' as the basis, we then vary K in the neighborhood of K', apply pLSA using each choice of K and compare the resulting extracted activity patterns to see whether they differ significantly, and select the value K that yields the most stable results (i.e., when increasing K further, the resulting extracted patterns stay nearly the same). This method is detailed below.

To determine the initial estimate K' using SVD, we employ the standard scree plot of the singular values: given a small threshold $\epsilon > 0$, we deem a singular value σ_k significant if $\sigma_k/\sigma_1 \geq \epsilon$, in which σ_1 is the largest singular value. Using this method, we apply SVD to HPAMs to determine k for different subnets within the campus network. Figs. 1(a)-(b) show the histogram of the resulting k's for all the subnets of size /24 for $\epsilon = 0.1$ and $\epsilon = 0.05$, respectively. We have validated that the low-rank approximations thus obtained capture at least 95% of the energy in the original HPAM (or, with squared errors ≤ 0.05).

Given the initial estimate K', we follow a fairly conservative approach to determine the "optimal" K. Starting with $K = \max\{K' - 2, 0\}$, we apply pLSA with each choice of K, and compare the resulting extracted activity patterns using the Jensen-Shannon distance (explained in Section IV). If the (minimum) distances between the extracted activity patterns using the previous value of K and those using the current

[1]There are other ways we may define s_h, as long as $\sum_{h \in H} s_h = 1$. For example, let $q_h = f_h/f_H$, and define $Q = -\sum_{h \in H} q_h \log q_h$, which is the *entropy* of the host flow distribution, $\{q_h, h \in H\}$. For each $h \in H$, define $s_h = (-q_h \log q_h)/Q$. Then s_h measures the ratio of h's entropy over the total entropy of the subnet. It provides another measure how "active" h is. Using such definition of s_h in eq.(6) would dampen the effect of one or two "outliers" that generate significantly more traffic than other hosts. We have applied pLSA using both versions of HPAM and the results are generally very similar, suggesting that we do rarely see dominating "outlier" hosts within each subnet.

HPAM top ports	aoSrcPort(0.234), dstTCP80(0.224), aoDstPort(0.124), srcTCP25(0.075), dstTCP113(0.056), srcTCP993(0.054), dstICMP0(0.023), srcTCP22(0.017), dstTCP25(0.015), srcICMP771(0.015)
HPAM top IPs	1(0.454), 33(0.059), 66(0.039), 14(0.027), 61(0.026), 106(0.023), 157(0.019), 35(0.017), 15(0.016), 87(0.016)
pLSA top ports $Pr(p\|c)$	aoSrcPort(0.500), dstTCP80(0.471), aoDstPort(0.233), dstICMP0(0.371), srcICMP771(0.347), srcTCP22(0.363), srcTCP25(0.168), dstTCP113(0.126), srcTCP993(0.120), dstTCP443(0.016)
pLSA top IPs $Pr(h\|c)$	1(1.00), 30(0.119), 33(0.104), 5(0.098), 71(0.086), 14(0.085), 66(0.083), 88(0.081), 15(0.077), 16(0.076)
Activity 1 ports	aoSrcPort(0.500), dstTCP80(0.471)
Activity 2 ports	aoDstPort(0.233), srcTCP25(0.168), aoSrcPort(0.164), dstTCP113(0.126), srcTCP993(0.120)
Activity 3 ports	aoDstPort(0.394), srcTCP22(0.363)
Activity 4 ports	dstICMP0(0.371), srcICMP771(0.347)
Activity 1 IPs	33(0.104), 14(0.085), 66(0.083), 15(0.077), 47(0.061), 61(0.056), 60(0.053), 157(0.051)
Activity 2 IPs	1(1.000)
Activity 3 IPs	30(0.119), 5(0.098), 71(0.086), 88(0.081), 16(0.076), 83(0.071), 6(0.063), 159(0.062), 35(0.056)
Activity 4 IPs	36(0.037), 50(0.035), 41(0.035), 20(0.034), 158(0.034), 65(0.034), 156(0.034), 141(0.034), 19(0.033), 84(0.032)

TABLE I
ACTIVITY PATTERNS FOR A /24 DEPARTMENTAL SUBNET

HPAM top ports	dstTCP80(0.479), aoSrcPort(0.233), aoDstPort(0.112), srcTCP6881(0.073), srcUDP6881(0.039), dstTCP5190(0.020), dstTCP1863(0.017), dstTCP443(0.017), dstTCP5050(0.006), dstTCP110(0.004)
HPAM top IPs	29(0.372), 19(0.096), 13(0.085), 146(0.075), 109(0.073), 74(0.066), 167(0.066), 188(0.060)-180(0.054), 149(0.053)
pLSA top ports $Pr(p\|c)$	dstTCP80(0.462), aoSrcPort(0.434), aoDstPort(0.391), dstTCP5190(0.278), srcTCP6881(0.277), dstTCP1863(0.227), srcUDP6881(0.147), aoDstPort(0.140), dstTCP5050(0.078) dstTCP110(0.017)
pLSA top IPs $Pr(h\|c)$	29(0.972), 74(0.489), 146(0.234), 188(0.200), 34(0.129), 81(0.108), 50(0.107), 19(0.102), 167(0.072), 37(0.069)
Activity 1 ports	dstTCP5190(0.278), dstTCP1863(0.227), dstTCP5050(0.078)
Activity 2 ports	dstTCP80(0.462), aoSrcPort(0.434)
Activity 3 ports	aoDstPort(0.391), srcTCP6881(0.277), srcUDP6881(0.147)
Activity 4 ports	aoSrcPort(0.397), aoDstPort(0.140)
Activity 1 IPs	146(0.234), 34(0.129), 81(0.108), 50(0.107)
Activity 2 IPs	19(0.102), 167(0.072), 180(0.067), 177(0.066), 197(0.051), 175(0.050)
Activity 3 IPs	29(0.972), 60(0.012)
Activity 4 IPs	74(0.489), 188(0.200), 37(0.069), 156(0.056)

TABLE II
ACTIVITY PATTERNS FOR A /24 WIRELESS SUBNET

value of K are less than a pre-specified threshold, we stop; otherwise, increase K by one and repeat. Using our network data, we find that this process always stops with $K \leq K' + 3$.

B. Illustration of Activity Patterns

In this subsection, we present examples of activity patterns extracted locally from subnets by applying pLSA to their HPAMs. Based on the significant host-port associations depicted in each activity pattern, we describe their underlying structures and demonstrate how each activity pattern represents a set of *coherent* applications (e.g. Email ports 993, 25 and 464, or HTTP ports 80 and 443). Moreover, we illustrate how the method can be utilized to compare, correlate and distinguish activity patterns providing an insight into how similar (or distinct) are two given subnets.

For each activity pattern c, we utilize the posterior probabilities $Pr(p|c)$ and $Pr(h|c)$ in Eq. 3 and Eq. 4, to obtain the set of ports and hosts corresponding to c, respectively. We, then, apply the entropy-based significant value extraction method developed in [16] to identify the *most significant* ports and hosts for each c (with $\beta = 0.9$).

Table III-A and Table III-A depict two examples of /24 subnets, one representing a departmental subnet and the other representing a wireless subnet. For each subnet represented by its HPAM, we list in the first 2 rows the top 10 ports as well as the top 10 IP hosts, respectively (based on the number of outgoing flows). The values in the parentheses represent the percentages of flows for each port/host over the total flows of the block. Similarly, in the third and fourth rows, we show the top significant $Pr(p|c)$ and $Pr(h|c)$ (by considering all $c_i \in [c_1 \cdots c_k]$). The rest of the rows list the activity patterns showing the significant ports as well as the hosts.

As explained earlier, the pLSA method extracts activity patterns reflecting the significant associations between ports and hosts. A significant association is characterized by a host or a number of hosts sending relatively large number of flows using some ports. Therefore, we expect ports as well as hosts associated with large number of outgoing flows in HPAM to be represented in the significant activity patterns. This is evident from the comparison of the set of top 10 ports in the (first row) and $Pr(p|c)$ (third row) which show that at least 9 of the 10 ports are identical. Similarly is the case of IP addresses comparing second and fourth rows. This observation stresses the fact that pLSA does not attempt to extract all possible host-port associations, but rather accurately captures associations that are dominant, significant and representative of large share of the block's host-port interactions.

The listed activities on the table further explain what these associations stand for, and the type of applications they represent. Table III-A lists 4 different activity pattern extracted from a departmental block. The first activity represents an HTTP client using *dst port 80* and *aoSrcPort*. HTTP client is the most frequently-used service in our campus network and is usually utilized by a large number of hosts. Therefore and as shown on the table, this activity pattern is associated with a relatively high number of hosts. In contrast, activity 2 represents an email server using SMTP *src port 25* and IMAP *src port 993* along with the IDENT *dst port 113* used to connect to

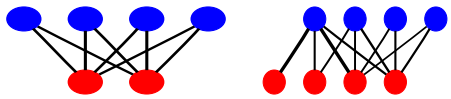
email servers along with *aoSrcPort* and *aoDstPort*. The use of *dst port 113* suggests that the local SMTP server accesses other outside email servers to exchange mail. Typical of such servers in which only one or few email servers exist within a block, this activity pattern is associated with one significant host (server). Similarly, activity 3 represents an SSH server along *src port 22* common on departmental blocks providing secure remote login. The difference between this server and the server in the previous activity pattern is that it corresponds to a set of machines making available SSH some of which are more heavily used than others. Activity 4 represents a totally different type of activity pattern corresponding to ICMP activity using ICMP *dst port 0* and ICMP *src port 771*. A number of hosts access this port which is likely a response to some incoming "ping" messages.

Unlike the departmental block, Table III-A represents activities in a (dynamic) wireless block. Activity 1 of this block represents Instant Messaging (IM) using AOL *dst port 5190*, MSN Messenger *dst port 1863* and Yahoo Messenger *dst port 5050*. IM service is popular among dynamic and residential blocks and as shown is accessed by a few hosts. Activity 2 is similar to activity 1 in the departmental block representing HTTP client using *dst port 80* and *aoSrcPort*. This service as mentioned earlier is popular and is accessed by a number of users. Common to residential and dynamic subnets, p2p and file sharing service are highly popular on such subnets which is depicted by activity 3 with BitTorrent *src UDP/TCP 6881*, in addition to *aoSrcPort* and *aoDstPort*. From these two ports, it seems that the hosts make available the access for others to download files. Activity 4 for this block is rather unique. The dominance of *aoSrcPort* and *aoDstPort* suggests there are a number of source and destination ports within the block none of which are significant, which implies that there are some server as well as client activities with low significance. This type of activity is likely to correspond to p2p traffic and further investigation on this block reveals that such source and destination ports are rather general-purpose ports which are not used for specific service but nonetheless mimic p2p traffic. From the table, this activity is associated with multiple hosts within the block.

The previous example demonstrates how pLSA can capture the different activities within the block extracting activity patterns that are *coherent* and represent a class of similar applications. Even though these two subnets are radically different (e.g. departmental vs. dynamic), we can still find commonly used activities found in both subnets. For example, HTTP client is the most dominant activity and not only both blocks contain this type of activity, but also exhibit the same structure in that both *port 80* and *aoSrcPort* are dominant connecting several hosts. Therefore, we can cluster similar activity patterns together and extract the dominant activity patterns within the network providing the big picture of the network as will be explained in details in section IV.

C. Inferring Network Applications from Activity Patterns

In the previous section, we demonstrated that the activity patterns capture the dominant activities within a particular subnet. In addition, these activity patterns also provide us with good amount of knowledge for inferring the specific applications associated with them. For example, an activity pattern with *aoSrcPort* being dominant is likely associated with client activities. In contrast, an activity pattern with *aoDstPort* being dominant often corresponds to server activities. If both *aoSrcPort* and *aoDstPort* are dominant, they are likely related to p2p activities.



(a) HTTP Server (b) Email Server

Fig. 2. Visualization of Server Activity Patterns

Besides these factious ports, the number of dominant source/destination ports and the number of dominant hosts associated with each activity pattern can also help us determine the specific application behind the activity patterns. As an example, we visualize and compare the interaction of the dominant hosts and dominant ports for activity patterns from the HTTP server activity and Email server activity using the host-port relation graph in Fig. 2. The red nodes in each graph represent the dominant hosts and the blue nodes stand for the dominant ports, which are both extracted following the method in Section III-B. An edge is drawn between a host h and a port p if at least one flow is observed from/to h towards/from port p. Though both of the activity patterns correspond to the server activity, the email server activity contains more dominant source ports than the HTTP server activity. For the HTTP server, the dominant ports are source *port 80* and *aoSrcPort*, while for the Email server, the dominant ports contain source *port 25*, source *port 993*, source *port 465* and *aoSrcPort*. We also observe distinct properties in other activity patterns, for example, the graph associated with the p2p activities often contain both dominant aoSrcPort and aoDstPort and one or a few other dominant ports depending on the specific applications (e.g., TCP 6346 for BitTorrent, 6881 for Gnutella and TCP 4662 for eMule) revealing that a p2p host behave as both a client and a server. In addition, we observe that the SSH server activity are is associated with more dominant hosts than other server activities, which is possibly due to the dominant SSH servers in the department networks. Though the above inference rules are quite intuitive, summarizing all such rules manually is a tedious and hard task. Therefore, we use a supervised machine learning technique for automatically learning these rules. We detail our approach below.

The samples in our application are host-port relation graphs and the labels (ground truth) are the associated application classes. Since most machine learning algorithms take samples in the format of feature vectors, we encode the host-port relation graphs into the feature vector representation as follows. For each graph, we compute 5 graph statistics: the number of dominant source ports, the number of dominant destination ports, the presence/absence of aoSrcPort and aoDstPort (binary feature) and the number of hosts connected to the graph. According to the network management need, we define 9 traffic classes: *HTTP Client*, *HTTP Server*, *Email Client*, *Email*

Size	AP	Ports
/24	1	dstTCP554, dstTCP80,aoSrcPort
	2	dstTCP80, aoSrcPort
	3	dstUDP500, dstUDP10000, srcUDP500, srcUDP10000
	4	srcTCP22, aoDstPort
	5	dstUDP53, srcUDP32906
	6	dstTCP80, dstTCP5190, dstTCP5222, dstTCP1863, aoSrcPort
	7	aoDstPort, srcTCP993, srcTCP25
	8	aoDstPort, srcTCP80
/20	9	srcTCP22, aoDstPort, srcUDP500, dstUDP500
	10	aoSrcPort, dstTCP1863, dstTCP5190, dstTCP5050
	11	dstTCP80, srcUDP47892
	12	srcICMP0, dstICMP0, srcTCP143, aoDstPort
	13	srcTCP80, aoDstPort
	14	aoSrcPort, dstTCP6667, srcUDP32773, dstTCP25
	15	dstUDP53, srcUDP32906, dstICMP0, srcICMP8
	16	aoDstPort, aoSrcPort, srcTCP21, srcTCP20
	17	aoDstPort, aoSrcPort, srcTCP25, srcTCP993
	18	srcTCP443, aoDstPort
	19	aoSrcPort, dstTCP80, dstTCP443, aoDstPort
/26	20	aoSrcPort, dstUDP123, dstTCP80
	21	aoSrcPort, dstTCP80, dstTCP5190, dstTCP143, dstTCP993
	22	aoDstPort, srcTCP80, srcTCP25, srcTCP22
	23	dstTCP80, aoSrcPort
	24	aoDstPort, dstICMP0, srcTCP80, srcICMP771
	25	dstTCP80, dstTCP443
	26	srcTCP3113, dstTCP5050, aoSrcPort
	27	aoDstPort, srcTCP22, srcTCP80

TABLE III
ACTIVITY PATTERNS FOR AN ENGINEERING SUBNET

Server, *Game*, *IM*, *p2p* and *SSH* and manually assign one of these classes to each feature vector. [2]. We then train a decision tree [17] to learn the association between feature vectors and the corresponding class labels. The decision tree is constructed using Weka [18] with all default parameters and evaluated through 10 folds cross-validation.

We show the decision tree offers a classification accuracy of 86.7% from the cross-validation results. The per-class accuracy varies from 100% for p2p activities to 58.3% for IM activities. We argue that our method is not designed for providing the most accurate traffic classification result, but to help the network administrator gain a fast understanding on the activities in the target subnet. When interesting activities or anomaly behaviors are observed in the subnet, the administrator can then apply more sophisticated, and hence more computationally expensive, method for further analysis.

D. Effect of subnet size

In the previous subsections and for illustration purposes, we have shown the results of our method using /24 subnets for which pLSA provides a *local* view of the activity patterns within a subnet. However, it is of interest to network and security analysts to gain a "global" view of the activity patterns for a larger subnet, or more generally, for the whole network. Consequently, we pose the question "What is the effect of the subnet size on the pLSA extraction results?".

Table III-C shows the results of applying pLSA to the same /24 subnet S using different block sizes /26 and /20 in which we list the activity patterns for each block in which activity patterns 1-8 correspond to applying pLSA to S, 9-19 correspond to applying pLSA to the /20 block containing S, and 20-27 correspond to applying pLSA to all the /26 blocks contained in S. From the table, we notice that the majority of the activity patterns extracted using the smaller /24 subnet are contained within the activity patterns extracted using the /20 subnet size in addition to newly extracted activity patterns 11, 12, 14, 16, 18 and 19 from the larger /20 block.

While activity patterns 8 and 13 are identical, activity patterns 6 and 10 representing IM activity as well as 7 and 17 representing Email server are similar.

However, we notice that activity patterns 3 and 4 have been merged into the single activity pattern 9 despite the fact that they are two distinct activity patterns with the first representing system management activity while the other corresponds to SSH server activity. Combining these two distinct activity patterns together resembles incoherent activity pattern that is not easy to interpret. This is similar to activity pattern 1 for the /24 block which describes a unique activity: real-time streaming over HTTP which is masked (no longer exists) within the /20 block activity patterns.

In contrast, each extracted activity pattern from the /26 subnets contains multiple (heterogeneous) activities. For example, activity pattern 21 represents IM using *port 5190*, HTTP client using *port 80*, and email client *using ports 143 and 993*. Similarly, activity pattern 22 combines HTTP server using *port 80*, SSH server using *port 22* and Email server using *port 25*. Therefore, using /26 block size seems to produce general activity patterns ignoring the distinct activities the ports represent within each activity pattern.

The previous example illustrates the impact of the subnet size on pLSA results. While extracting significant activity patterns from a larger subnet (e.g. by observing traffic at some *central* vantage point) may provide a global view of the activity patterns within the network, this method has major drawbacks. First, the unscalability and complexity of extracting activities from a large network may render such method unfeasible. Second, while the method provides a high-level picture of the significant activities within the network, it might mask important patterns that would have otherwise been discovered from a smaller subnet. Third, the power of extracting activity patterns from a subnet lies on the interpretability of the results in which the activity patterns are coherent and can be precisely described; an endeavor that may be unattainable for larger subnets as described above. On the other hand, as we decrease the subnet size (and hence decrease the number of hosts within each subnet), there are less active hosts within the smaller /26 blocks causing multiple activity patterns to merge into one or two activity patterns that contain heterogeneous activity patterns. Moreover, since hosts involved in the same type of activity are within different /26 blocks but within the same larger /24 block, their host-port associations are scattered in different /26 blocks and cannot be realized as significant unless we extract a larger block size.

We have tried different block sizes and found that blocks between the sizes of /24 and /21 provide similar results. In this section, we presented the first stage of the local-global two-stage process in which we applied pLSA to small subnet sizes

[2]These labeled activity patterns cover 62% of all the activity patterns. Validation on the inference results for the unknown (unlabeled) activity patterns is left for future work.

(e.g. /24 blocks) to extract the significant activity patterns.

However, in order to provide a *global* view of the activity patterns for the whole network, in the next section, we employ the second stage in which we perform a global classification of activity patterns analyzing the similarities of locally extracted activity patterns and grouping them accordingly using a hierarchical clustering algorithm. In addition to obtaining the "big-picture" of the network-wide activity patterns, the advantage of this method is that it can be *distributed* which resolves scalability and complexity issues encountered when applying pLSA to larger subnets.

IV. Clustering activity patterns globally

In the previous section, we provided the detailed technique for the first stage of our HEAP's two-stage method in which we utilized pLSA to extract locally significant activity patterns for individual subnets. In this section, we zero in the second stage of HEAP in which we perform a *global* clustering of activity patterns obtained from the first stage.

In subsection IV-A, we present our novel methodology. First, we utilize the *Jensen-Shannon distance (JSD)* metric to compare the activity patterns extracted from individual subnets. We then provide a hierarchical clustering algorithm to cluster similar activity patterns which compares and merges clusters based on the average JSD of activity patterns. The advantage of the two-stage clustering method is its efficiency and scalability which is realized by its distributive manner in which pLSA is applied to individual subnets and the resultant activity patterns are subsequently grouped based on their similarities using hierarchical clustering. In subsection IV-B, we present our clustering results in which we distinguish clusters (i.e. types of activity patterns) that have network-wide dominance from clusters prevalent to individual subnets.

A. Hierarchical Clustering of Activity Patterns

In pLSA, each activity pattern, c, is characterized by its port and host posterior probability distributions, $Pr(p|c)$ and $Pr(h|c)$, respectively. More specifically, $Pr(p|c)$ describes the underlying applications for c. Therefore, comparing two activity patterns c_1 and c_2 involves comparing their corresponding posterior probability distributions $Pr(p|c_1)$ and $Pr(p|c_2)$, respectively.

Such comparison can be computed by utilizing the Jensen-Shannon distance (*JSD*), which measures the distance between two probability distributions. JSD computes the distance between two activity patterns as follows. Given two activity patterns c_1 and c_2 represented by their port probability distribution P_1 and P_2, respectively, the Jensen-Shannon distance $djs(P_1, P_2) = \frac{1}{2}d(P_1||M) + \frac{1}{2}d(P_2||M)$ where $M = \frac{1}{2}(P_1 + P_2)$ and $d(P||Q)$ is the Kullback-Leibler divergence $d(P||Q) = \sum_{i=1}^{n} P(i)log\frac{P(i)}{Q(i)}$.

Using the above definition, we compute the pairwise distance between all pairs of activity patterns, and the results are shown on Fig 3: the rows and columns are indexed by the activity patterns, and a gray scale is used to visually depict the similarity: the darker a point (i, j) is, the shorter is the distance between the two activity patterns (c_i, c_j). The rows and columns are sorted so that the activity patterns with likely similar behavior (i.e. running same applications), are located closer to each other. The figure clearly shows clusters of activity patterns with likely similar behaviors.

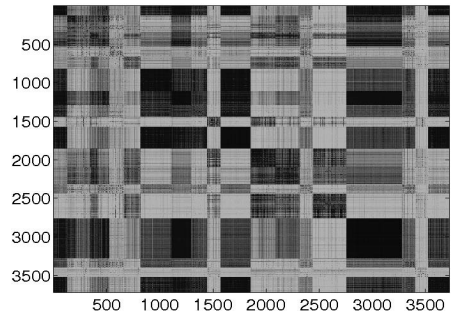


Fig. 3. Activity patterns similarity matrix

To extract plausible clusters shown on Fig 3, we apply a hierarchical clustering algorithm. Represented as a tree structure, hierarchical clustering provides the flexibility of choosing clusters at different levels of granularity which is made available through cutting the tree at a certain level. For example, we may want to cluster more generic activity patterns (such as HTTP, Email or p2p) to provide a high level of the dominant activities within the network. Alternatively, it may be desirable to obtain fine-grained clustering details, (e.g. separating p2p into Gnutella, eMule and BitTorrent). Both cases are realized by cutting the tree at different levels of granularity.

In hierarchical clustering, a node (activity pattern or cluster of activity patterns) is represented by its probability distribution $Pr(p|c)$. The algorithm iteratively merges points based on their pairwise JSD values. In each iteration, the two points with the lowest JSD are merged together into a cluster and $Pr(p|c)$ is (re)computed by taking the average of all $Pr(p|c_i)$ within the cluster. This iterative process is continued until all activity patterns have been merged into one cluster.

B. Clustering Results

In hierarchical clustering, the results can be expressed in the form of a tree structure. For this tree, the leaf nodes represent activity patterns and non-leaf nodes represent clusters of activity patterns. The closer are the nodes (i.e. share relatively short-path to a common ancestor), the more similar they are. Hence, hierarchical clustering is useful for our case, since it enables us to conduct a "multi-scale" clustering at different levels of granularity. Therefore, we can provide coarse-grained clusters (e.g. p2p) or fine-grained clusters (e.g. Gnutella, BitTorrent and eMule).

To distinguish network-wide activity patterns from activity patterns prevalent to selected blocks, we obtain a number of representative clusters for each of them. For interpretability, we require the obtained clusters to exhibit a high level of "coherence", i.e. activity patterns within the same cluster have smaller JSD (computed as the average JSD of all activity patterns within the cluster). A cluster is "coherent" and thus interpretable if all activity patterns within it have relatively small pairwise JSD. Fig 4 shows the relative increase in JSD

No	Label	Ports	Subnets	%	NoAPs
1	HTTP Client	dstTCP80, dstTCP443, aoSrcPort	Network-wide	40.8	1520
2	IM	dstTCP5190, dstTCP1863, dstTCP5050, dstTCP5222, aoSrcPort	Network-wide	1.3	50
3	HTTP Server	srcTCP80, srcTCP443, aoDstPort	Departments	7.2	268
4	SSH Sever	srcTCP22, aoDstPort	Departments	1.2	48
5	Email Server	srcTCP25, srcTCP993, srcTCP143, srcTCP465	Different Academic Departments	0.9	34
6	Non-Significant Ports	aoSrcPort, aoDstPort	Network-wide	0.9	34
7	Diverse Client Activity	aoSrcPort	Different Academic Departments	0.8	31
8	Diverse Server Activity	aoDstPort	Different Academic Departments	0.5	17
9	ICMP	dstICMP0, srcICMP0, srcICMP771	Network-wide	2.1	77
10	Time Synchronization	srcUDP123, dstUDP123	Different Academic Departments	0.9	32
11	DNS	dstUDP53, aoSrcPort	Several Academic Departments	0.3	13
12	mySql	srcTCP3306, aoDstPort	Different Academic Department	0.4	14
13	Gnutella	{dst,src}{TCP,UDP}6881, aoSrcPort, aoDstPort	Residential Halls	0.7	27
14	BitTorrent	{dst,src}{TCP,UDP}{6346,6348}, aoSrcPort, aoDstPort	Residential Halls	0.5	20
15	eMule	{dst,src}{TCP,UDP}{4662, 4672}, aoDstPort, aoSrcPort	Residential Halls	0.3	11
16	IRC	dstTCP7000, dstTCP6667, aoSrcPort	Residential Halls	0.3	12
17	Admin	{dst,src}{TCP,UDP}{500,10000}	Mostly CS Blocks	0.1	5
18	GoToMyPC	dstTCP8200, aoSrcPort	Medical School	0.4	14
19	Printer Sharing	srcTCP631, aoDstPort	U Services	0.1	2
20	Telnet	dstTCP23, aoSrcPort	Secure Wireless Blocks	0.1	2
21	Scanning	srcTCP1047, srcTCP1046, srcTCP1045, dstTCP80	Different Departments	0.1	2

TABLE IV
CLUSTERING RESULTS

at each step of the iterative algorithm in which the x-axis represents the activity pattern being merged and the y-axis represents the relative increase in JSD at each merging step. As shown on the plot, each merging step incurs insignificant increase in JSD until we reach a point in which a sharp increase is incurred. The actual JSD value corresponding to the merging step just prior to the sharp increase is 0.25.

Table IV-A shows sample clusters with $JSD \leq 0.25$. On the table, we list clusters of network-wide activity patterns as well as clusters of activity patterns prevalent to specific blocks.

HTTP client activities are the most dominant network-wide activities. Depending on the HTTP site being accessed, the port being used is either unsecure TCP dstPort 80, secure TCP dstPort 443 or both (e.g. redirection from an unsecure site to a secure site) which implies that a block may have more than one HTTP client activity. HTTP client activities may correlate with other types of activities, such as Instant Messaging (IM) in which the large share of the activity corresponds to HTTP client and a much smaller share, yet significant, of the activity corresponds to IM ports. However, there are some activity patterns representing only IM traffic (as shown in cluster 2) which are more prevalent to wireless and residential halls as users utilize more than one IM service.

There are a few server activities including HTTP server (cluster 3) widely deployed in departmental subnets with one or few servers, SSH servers (cluster 4) which usually contains a number of SSH servers within the block providing secure remote login service, and Email servers (cluster 4) deployed in selected subnets, such as large academic and wide-campus service departments (e.g. OHR, IT..etc) containing one or two servers within the block.

Cluster 6 contains a number of server as well as client activity patterns none of which are significant. Hence the dominant ports are *aoSrcPort* and *aoDstPort*. Such activity pattern is typical of p2p activities. Similar to this activity pattern is cluster 7 which corresponds to a large number of insignificant client activities. (Similarly is cluster 8 of large number of insignificant server activities).

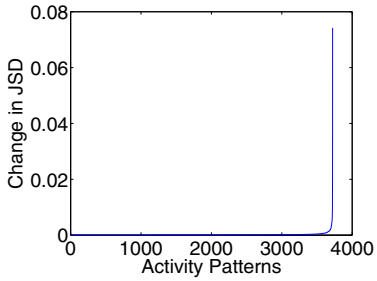


Fig. 4. Relative increase in JSD at each merging step

The ninth cluster corresponds to ICMP traffic with relatively large number of activity patterns, which may be a sign of queries generated to confirm the existence of some type of service which may be (potentially) malicious. Cluster 10 represents network time protocol; a utility commonly used in networks to resist the effect of latency which utilizes time synchronization (e.g. with ISPs) while cluster 11 represents reverse DNS lookup for which there are usually few host responsible for such queries. Several blocks may make available databases to provide information to the academic community. Cluster 12 is an example of such activity pattern in which several academic departments provide information via mySql server. An example of this is a biology-related department providing genes' information in a database available as public knowledge.

Clusters 13, 14 and 15 correspond to p2p traffic, including Gnutella, BitTorrent, and eMule, respectively. Used for file sharing, these activity patterns include *aoSrcPort* and *aoDstPort* as dominant ports and are usually associated with residential halls blocks in which several hosts utilize these types of services. A similar activity is that of cluster 16 which corresponds to chatting that is popular in residential halls.

The aforementioned clusters contain activity patterns common for the network-wide activities some of which are general to all types of subnets (HTTP Client) and others specific to one

type of subnets (e.g. p2p in residential halls). In addition, there are activity patterns which are prevalent to specific subnets. For example, cluster 17 corresponds to engineering blocks containing hosts which connect to outside hosts for software and tools automatic update. Cluster 18, on the other hand, is utilized by hosts within the medical school allowing users login to remote (off-campus) machines (e.g. machines for special service or their own PCs) using a specific *dst port 8200* which belongs to the GoToMyPC service. Some special-purpose protocols are used mainly by the campus support services such as that of cluster 19 for utilizing printer sharing using the Internet Printing Protocol (IPP). Telnet and scanning (clusters 20 and 21) are other types of activity patterns that are prevalent in select blocks. Due to space limitation, we only list a few types of clusters of activity patterns prevalent exclusively to specific subnets.

Our results allow us to distinguish clusters common to specific blocks from those that are common to the whole network. The results also depict the common activities to be found in a typical campus network. Therefore, we are provided with a "global-view" of what is going on within the campus network. In the next section, we go one step further and profile and label each subnet based on its underlying activity patterns. While our clustering results provide more generic labels of activities based on the threshold of $JSD = 0.25$, one can obtain finer grained clusters by propagating further down the tree to retrieve such clusters.

V. Tracking Block Behaviors over Time

In this section, we demonstrate how to track the change of block activities over time. The stability of the block activities often reveals the composition of the hosts within the block and the major activity change in the block can be an indicator of potential attacks or other anomalies.

A. Stability of Blocks

Given a block B, let $V^{(t)} = [v_1, \ldots, v_K]$ denote the set of its activity patterns extracted using pLSA at the tth time interval (say, the tth day). Similarly, let $V^{(t+1)} = [w_1, \ldots, w_L]$ be the activity patterns corresponding to the same block at the $t+1$th time interval. We note that $K \neq L$ in general, since new activities may show up at $t+1$ and, meanwhile, old activities may disappear. Even for some blocks when $K = L$, the activity patterns may still change significantly. Therefore, we define *block activity deviation (DEV)* for measuring the difference of all activity patterns at time t and $t+1$.

$$DEV = \frac{\left(\sum_{i=1}^{K} JSD(v_i, w_i^*) + \sum_{j=1}^{L} JSD(v_j^*, w_j) \right)}{K + L} \quad (7)$$

where $w_i^* = argmin_{w_j} JSD(v_i, w_j)$, $1 \leq j \leq L$, which represents the activity pattern at $t+1$ that is most similar to v_i in terms of JSD. Similarly, $v_j^* = argmin_{v_i} JSD(v_i, w_j)$, $1 \leq i \leq K$. DEV measures the average JSD of each activity pattern appearing at t (resp. $t+1$) to its "best matching" peer at $t+1$ (resp. t). If the activity patterns have no intersection at t and $t+1$, DEV achieves the maximal value of 1. In contrast, if all the activity patterns remain the same, DEV equals 0.

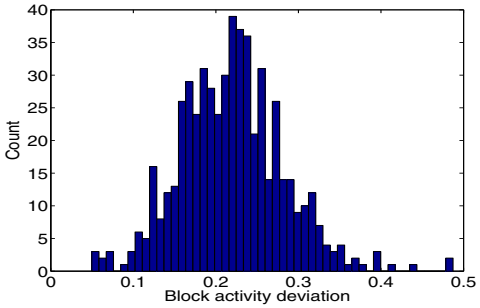


Fig. 5. Block activity deviation between 02/08/2006 and 02/09/2006.

Choosing $t = 02/08/2006$ and $t + 1 = 02/09/2006$, we show the DEV's for all the class C blocks in Fig. 5. The distribution of DEV's is characterized with a bell shape, with most (90%) of the blocks having DEV between 0.1 and 0.3. We interpret the stability of individual blocks based on the domain names of the hosts within the block and the network traffic observed in the block.

The 10 most stable blocks (with the lowest DEV's) are all department blocks or library blocks. Each of these blocks contains multiple server machines, like Email servers, HTTP servers and SSH servers, etc. In addition to the server activities, these blocks also contain limited client activities like HTTP, Email, SSH and IM. We conjecture this is likely due to the fact that the machines in these blocks are configured to only provide limited functionalities according to the department policies. In contrast, when the DEV becomes higher, we start seeing blocks from the residential hall networks and dynamic blocks, such as wireless blocks and dialup blocks. In addition, a greater variety of traffic also appears, such as p2p and Game. The top 10 most unstable blocks (with the highest DEV's) all belong to dynamic blocks with domain names containing dhcp, wireless, and dialup. Most client activities vary significantly in two consecutive days. Take a wireless block as an example, at time t, the Web and IM client activities are more dominant in the block; however, at time $t+1$, the most dominant client activities become p2p with port TCP6881 (BitTorrent). More interestingly, we find some server activities appear temporally in these blocks, such as HTTP server activity (TCP80) and other server activities with unpopular ports (e.g., TCP1494 and UDP32459). Since according to the block assignment, we do not expect any server activity in these blocks, such server activities are likely to indicate suspicious or anomalous network behaviors, such as back-door attacks, etc. In the next section, we further explore this observation and design a method for identifying anomaly behaviors by tracking the temporal changes of network blocks.

B. Anomaly and Attack Detection.

We define an anomaly activity as the traffic activity that is different from the typical activities in a particular network block. For example, in a client-centered block, the appearance of a server activity often indicates back-door attacks. In this section, we propose a method based on the pLSA decomposition result for detecting anomalies in a particular network

block. We demonstrate the effectiveness of the proposed method through *attack emulation*, where we inject certain types of attacks or other anomalous activities into a block with otherwise "normal" activities. We have performed this study using a range of anomalies with a variety of activity patterns, including *outside scanning*, *back-door trojan activities* and *ddos attacks*. We emulate the outside scanning activity by injecting with 3 response flows from TCP srcPort25 to all active hosts in the network. To emulate the back-door trojan activities, we inject 1000 response flows associated with TCP srcPort 443 to one client machine in the network. We also emulate the ddos attacks by injecting 1000 echo reply flows (ICMP srcPort 0) to one server machine in the network.

When an anomaly occurs, if it is significant (either involving a large number of flows, e.g., ddos attacks, or involving a large number of hosts, e.g., scanning attacks), the pLSA decomposition result is expected to contain a new activity pattern corresponding to the anomaly behavior. Therefore, our goal is to identify the activity pattern which, after attack injection, deviates the most from all activity patterns before injection. Hence, we define an *anomaly score (AS)* as follows ($[v_1, \ldots, v_K]$ and $[w_1, \ldots, w_L]$ represents the sets of activity patterns before and after attack injection, respectively).

$$AS = \max_{1 \le j \le L} JSD(v_j^*, w_j) \tag{8}$$

where $JSD(v_j^*, w_j)$ is the minimum JSD between w_j and any of the activity patterns, or the "best matching" peer, before injection. The AS score ranges from 0 to 1, with 1 indicating the new activity is completely different from all activities before injection.

We select three different types of blocks for the attack emulation: 1) a departmental block which contains mostly Email servers and HTTP servers; 2) a residential hall block consisting mostly of client machines and 3) a very dynamic wireless block. We conduct the attack emulation on all these three blocks. As a baseline for comparison, we compute the AS scores for block activities between 02/08/2006 and 02/09/2006, which can be considered as the maximum possible normal activity change in the block. We summarize the results in Table V-B. The second column shows the normal change measured by the AS scores. We observe that for all the blocks, the most significant change of the normal activities only results in an AS score of 0.3-0.5. However, after injecting a significant anomaly activity which is different from all activity patterns in the network, the AS scores rise to 0.65-0.80. This suggests that our pLSA decomposition results indeed capture the emerging anomaly activity patterns in the network. Though the administrator needs to apply more complicated methods, e.g., deep packet inspection, and other source of information, e.g., port numbers or domain names, our method, due to its light-weight and high efficacy, can be used as a real-time tool for tracking the network block behaviors and identify potential anomaly activities in the network.

Description	Normal	Scanning	DDoS	Back-door
Departmental	0.4626	0.6553	0.6873	0.5835
Residential Hall	0.5280	0.7685	0.7616	0.6785
Wireless	0.3023	0.8312	0.8588	0.8013

TABLE V
ATTACK EMULATION AND ANOMALY DETECTION RESULT

VI. CONCLUSIONS

In this paper, we developed HEAPs, a scalable and effective methodology which employs a local-global two-stage process to automatically extract activity patterns from massive network data. In the first process, we illustrated the use of pLSA to extract locally significant activity patterns using network data from individual network subnets. Using the significant activity patterns extracted from individual subnets, in the second stage, we performed global classification of activity patterns by analyzing the similarities of locally extracted activity patterns and grouping them accordingly using a hierarchical clustering algorithm. We also demonstrated how HEAPs can be used to profile and track what are running on subnets. To further illustrate the utility of HEAPs, we artificially injected "attack" traffic into the real network data and showed how they could be detected through extracted anomalous activity patterns on individual subnets. Our method is general and scalable and can be applied to any other campus networks or enterprise network regardless of its size.

REFERENCES

[1] Thomas Hofmann. Unsupervised learning from dyadic data. pages 466–472. MIT Press, 1998.

[2] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. In *Machine Learning*, page 2001, 2001.

[3] Thomas Landauer, P. W. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, (25):259–284, 1998.

[4] F. De la Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1-3):117–142, 2003.

[5] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of KDD*, 2001.

[6] D. Lee and H. Seung. Learning the parts of objects. by non-negative matrix factorization. In *Nature*, 1999.

[7] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proc. of SIGIR*, 2003.

[8] C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proc. of ACM KDD*, 2006.

[9] David M. Blei, Andrew Y. Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, (3):9931022, 2003.

[10] P. Haffner, S. Sen, O. Spatscheck, and D. Wang. ACAS: Automated construction of application signatures. In *Proc. of MineNet*, 2005.

[11] T. Karagiannis, A. Broido, M. Faloutsos, and K. claffy. Transport layer identification of p2p traffic. In *Proc. of ACM IMC*, 2004.

[12] A. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. In *Proc. of ACM SIGMETRICS*, 2005.

[13] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti. Traffic classification using clustering algorithms. In *Proc. of the ACM SIGCOMM workshop on Mining network data (MineNet'06)*, 2006.

[14] E. Sharafuddin, Y. Jin, N. Jiang, and Z.-L. Zhang. Know your enemy, know yourself: Block-level network behavior profiling and tracking. In *Preprint*, 2009.

[15] Y. Jin, E. Sharafuddin, and Z-L. Zhang. Unveiling core network-wide communication patterns through application traffic activity graph decomposition. In *Proc. of SIGMETRICS '09*, pages 49–60, 2009.

[16] K. Xu, Z.-L. Zhang and S. Bhattacharyya. Profiling Internet backbone traffic: behavior models and applications. In *Proc. of ACM SIGCOMM*, August 2005.

[17] R. Duda, P. Hart, and D. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[18] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. 1999.