ARTICLE IN PRESS

Computer Networks xxx (2011) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Session level flow classification by packet size distribution and session grouping

Chun-Nan Lu [a,*], Chun-Ying Huang [b], Ying-Dar Lin [a], Yuan-Cheng Lai [c]

[a] Department of Computer Science, National Chiao Tung University, HsinChu, Taiwan
[b] Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung, Taiwan
[c] Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan

ARTICLE INFO

Article history:
Received 23 February 2010
Received in revised form 16 January 2011
Accepted 9 September 2011
Available online xxxx

Keywords:
Flow classification
Session grouping
Session classification
Packet size distribution

ABSTRACT

Classifying traffic into specific network applications is essential for application-aware network management and it becomes more challenging because modern applications complicate their network behaviors. While port number-based classifiers work only for some well-known applications and signature-based classifiers are not applicable to encrypted packet payloads, researchers tend to classify network traffic based on behaviors observed in network applications. In this paper, a session level flow classification (SLFC) approach is proposed to classify network flows as a session, which comprises of flows in the same conversation. SLFC first classifies flows into the corresponding applications by *packet size distribution* (PSD) and then groups flows as sessions by *port locality*. With PSD, each flow is transformed into a set of points in a two-dimension space and the distances between each flow and the representatives of pre-selected applications are computed. The flow is recognized as the application having a minimum distance. Meanwhile, *port locality* is used to group flows as sessions because an application often uses consecutive port numbers within a session. If flows of a session are classified into different applications, an arbitration algorithm is invoked to make the correction. The evaluation shows that SLFC achieves high accuracy rates on both flow and session classifications, say 99.9% and 99.98%, respectively. When SLFC is applied to online classification, it is able to make decisions quickly by checking at most 300 packets for long-lasting flows. Based on our test data, an average of 72% of packets in long-lasting flows can be skipped without reducing the classification accuracy rates.

1. Introduction

Classifying traffic into specific network applications is essential for application-aware network management. According to the classification results, an enterprise or a service provider can apply various rules to protect network resources or enforce organization policies. Accurate traffic classification is therefore the keystone in application-aware network management. However, it is not trivial to correctly classify the traffic into the applications according to their diverse characteristics and behaviors because traffic can be encrypted, relayed by other protocols, or disassembled.

A number of approaches have been proposed to identify and to classify the traffic into the applications. However, traditional classification methods may not work well for emerging application because they usually rely on either port numbers [1,2] or payload signatures [3–6]. To bypass policies enforced by network administrators, modern applications use several different techniques to make their network traffic invisible to network monitors. Common communication protocols, like HTTP, are often used as covert channels to relay other types of traffic. Both payload

* Corresponding author.
E-mail address: cnlu@cs.nctu.edu.tw (C.-N. Lu).

1389-1286/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2011.09.007

encryption and port randomization techniques are also adopted to increase difficulties for traffic classification. As a result, researchers now tend to classify traffic based on application behaviors. They monitor and model application behaviors, and then use the resultant application profiles to classify traffic.

Classifying traffic into applications becomes more challenging because of more sophisticated application behaviors. The connection behavior of one application may be similar to that of another application. For example, the behavior of an HTTP file transfer may look similar to that of an FTP one. In addition, not all flows generated in one session do the same thing. For example, a BitTorrent client may simultaneously establish several flows to retrieve the list of servers, look up resources, check peer status, and transfer files. Thus, to have a better classification result, we propose an approach, namely *session level flow classification* (SLFC), to classify network flows as sessions and hence obtain a complete picture of application behaviors.

SLFC contains two parts, i.e., flow classification and flow grouping. The former classifies flows into applications by *packet size distribution* (PSD) and the latter groups related flows as sessions by *port locality*. A flow is identified by the five-tuple information, which includes source IP, destination IP, source port, destination port, and protocol. When the PSD of one flow is determined, it is compared with each representative of all pre-selected applications to decide which application it should be. Since the information of packet payloads is not required, this method works even if the packet payloads are encrypted. In addition, flows will be grouped as sessions by checking port locality because operating systems often allocate consecutive port numbers for an application to setup connections with remote hosts. If the source and destination IP addresses of two flows are the same and their port numbers are consecutive, the two flows may belong to the same session of an application. If flows of a session are classified as different applications, an arbitration algorithm based on majority votes is invoked to make the correction. Evaluations and online benchmarks show that SLFC is able to obtain accurate results and make decision by checking at most 300 packets and the overall throughput exceeds 400 Mbps in a mainstream computer.

This rest of this paper is organized as follows. In the next section, some important related literatures are surveyed. Section 3 describes two basic observations, which are the base of designing our classification algorithm. The SLFC algorithm is formally presented in Section 4 and its performance is evaluated in Section 5. Finally, conclusions are given in Section 6.

2. Related work

Classifying network flows by using statistical properties of network traffic is not new. Such methods assume that the statistical properties of traffic are unique for different applications and can be used to distinguish applications from each other. The commonly used statistical features, for example, contain flow duration, packet inter-arrival time, packet size, bytes transferred, number of packets, and etc. Earlier work just focused on the characteristics of network traffic classes or applications. Paxson studied the relationship between statistical properties of flows and applications that generate them based on Internet traffic characterization [7] and Paxson and Folyd [8]. Paxson [7] modeled and analyzed the individual connection characteristics, such as bytes transferred, duration, arrival periodicity for different TCP applications. In [8], they found that user-initiated events, such as TELNET connections or FTP control connections arrivals, can be described by a Poisson process, whereas other connection arrivals deviate considerably from Poisson. They did not further attempt to classify network traffic according to different application layer protocols. These work, however, showed that it is possible to identify network traffic based on statistical features.

Hereafter, more work endeavored to classify exclusively network traffic based on statistical features. They generally consist of two parts: model building and classification. A model is first built using statistical attributes of flows by learning the inherent structural patterns of datasets and the model is then used to classify other new unseen network traffic. Dewes et al. [9] analyzed and classified different Internet chat traffic using multiple flow characteristics such as flow duration, packet inter-arrival time, packet size, and bytes transferred. Roughan et al. [10] used nearest neighbor (NN) and linear discriminant analysis (LDA) to map applications to different quality of service classes using features such as average packet size, flow duration, bytes per flow, packet per flow, and root mean square (RMS) packet size. Divakaran et al. [11] identified different classes of applications by observing packet train length and packet train size. A packet train is the flow of packets between two hosts in a network, where each packet forms the car of the train. Their approach is effective to classify short UDP flows, e.g., DNS traffic. However, when it is applied to long-lasting flows or TCP flows, this approach often makes incorrect decisions. Bernaille et al. [12] identified applications based on packet sizes and directions of packets. Application behavior is clustered by characteristics observed in the very first five packets of TCP connections. Then, a flow is classified into an application by measuring the minimum similarity distance. However, the solution can not handle either packet losses or packet reordering. Ying-Dar et al. [13] used packet size distribution (PSD) and packet size change cycle of a flow to model and classify application flows. Without the auxiliary mechanism, port association, the average accuracy rate is not good enough for online classification. Although our SLFC uses similar statistical features, say packet size distribution, SLFC is simpler and can achieve higher accuracy rates and better online speedup effect without human intervention.

A few works analyze traffic at a level other than flow level. Kannan et al. [14] used a connection-level trace to derive abstract descriptions of the session-structure for different applications present in the trace. Based on flows' statistical information, the Kannan's approach discovers and characterizes flow/session causality relationship. It can further infer applications' internal session structures.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

However, it may be not able to handle modern sophisticated applications since it identifies applications by using only port numbers.

BLINC, proposed by Karagiannis et al. [15], introduces another type of approach for traffic classification based on the analysis of host behavior. It associates Internet host behavior patterns with one or more applications, and refines the association by heuristics and behavior stratification. It is able to accurately associate hosts with the service they provide or use by inspecting all the flows generated by specific hosts. However, it can not classify a single TCP connection because it has to gather information from multiple flows for each individual host before it can decide on the role of the host.

Some alternative proposals [17–20] utilize machine learning (ML) techniques to network traffic, which are different from mathematical frameworks. ML is known as a collection of powerful techniques for knowledge discovery and data mining domains. The idea of applying ML techniques for traffic classification was introduced in [16]. ML techniques are divided into two phases as well. They first use similar statistical features, like aforementioned work, to build models but then apply particular ML techniques, dissimilar with aforementioned work, to classify network traffic. Different ML techniques may perform differently toward distinct network applications, and may require distinct parameter configurations. Therefore, McGregor et al. [17] used expectation maximization technique, Zander et al. [18] used AutoClass technique, and Moore [20] used Naïve Bayes technique, to group flows based on a set of flow statistics to classify traffic under different metrics and criteria.

3. Features utilized By SLFC

In this section, the two major features utilized by SLFC, i.e., the packet size distribution (PSD) and the port locality are introduced. Our observations show that application behaviors can be differentiated with their PSDs, meaning that flows of the same application have similar PSDs, but flows of different applications have diverse PSDs. Our observations also show that the port numbers used by flows belonging to the same application session are often adjacent. In this paper, a session is defined as a set of flows that are generated in the same conversation. For client–server applications, a session is defined as a single flow established between a client and a server. For peer-to-peer applications, a session is defined as several flows generated consciously in a peer-to-peer transaction.

3.1. Packet size distribution (PSD)

The PSD of a network application can be obtained from all its flows. We manually capture traces of a single application in a crafted environment to collect the traffic of a specific application. The major advantage of manual collection is that all collected traffic belongs to the same application. Each pre-selected application is executed in turn, and the traffic generated is recorded when it passes through the network interface. Table 1 lists the corresponding application names, application-level protocols, and transport layer protocols used of all pre-selected applications.

Table 1
Pre-selected applications.

Application name	Application-level protocol	Transport layer protocol
BitTorrent	P2P	TCP/UDP
eMule	P2P	TCP/UDP
Skype	P2P	TCP/UDP
HTTP	HTTP	TCP
POP3/SMTP	POP3/SMTP	TCP
FTP	FTP	TCP
ShoutCast	Streaming	TCP/UDP
PPLive	P2P Streaming	TCP/UDP

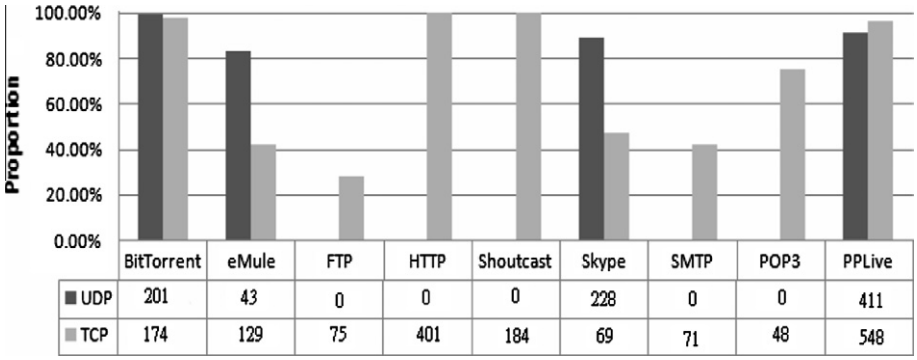
Different applications produce unequal packet sizes due to different operational requirements. Fig. 1 shows the most frequently used packet size of each pre-selected application except packets without payloads and packets with size of maximum transmission unit (MTU) in a normal transaction. Zeroes in the entries means that the applications have no traffic belonging to that protocol. Fig. 2(a), (b), and (c) show the use of packet sizes of three different applications, Shoutcast, FTP, eMule, respectively. The horizontal axis is the packet sequence number and the vertical one is the corresponding packet size. Fig. 2 presents that different applications have different PSDs. In addition, the packets of the same application have similar size distributions, as shown in Fig. 3(a) and (b), which present the packet size distributions of two BitTorrent instances. These observations demonstrate that PSD is a good feature to classify network traffic.

In the execution period of an application instance, packets generated can be roughly divided into two types, one of which is control packet and the other is data packet. Control packets are indispensable and mostly used for account authentication, information exchange initialization and status checking; while data packets are used for true data transfer. Even if packet sizes are variant and diverse for different applications, there must be some invariant or limited-variance control packets generated in the execution period, which are our primary targets.

3.2. Port locality

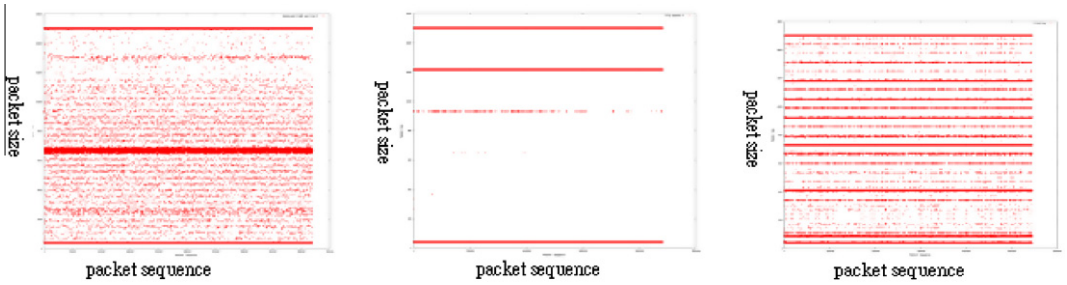
We observed that port numbers used by network flows of the same session often have the property of spatial locality, i.e., the port numbers are consecutive or very close to each other. Although port numbers may be randomly chosen, operating systems often allocate consecutive port numbers when an application has to setup several connections with remote hosts. This phenomenon is useful because when a flow is classified as one specific session, the port numbers can be used to associate flows the same session. Although port locality is useful, some operating systems do not follow the common rule and hence it is not always able to associate flows as a session by using port locality. In this case, a single flow is treated as a session. Fig. 4 shows the port numbers used by flows of multiple HTTP sessions. Port numbers the servers used are not

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007



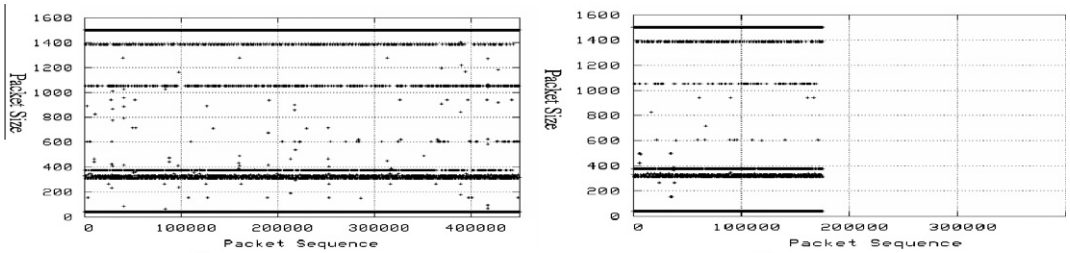
	BitTorrent	eMule	FTP	HTTP	Shoutcast	Skype	SMTP	POP3	PPLive
UDP	201	43	0	0	0	228	0	0	411
TCP	174	129	75	401	184	69	71	48	548

Fig. 1. Different applications have different most frequent packet size.



(a) PSD of one Shoutcast instance (b) PSD of one FTP instance (c) PSD of one eMule instance

Fig. 2. Different types of applications have distinct PSD.



(a) BitTorrent Instance 1 (b) BitTorrent Instance 2

Fig. 3. Two instances of BitTorrent have similar PSD.

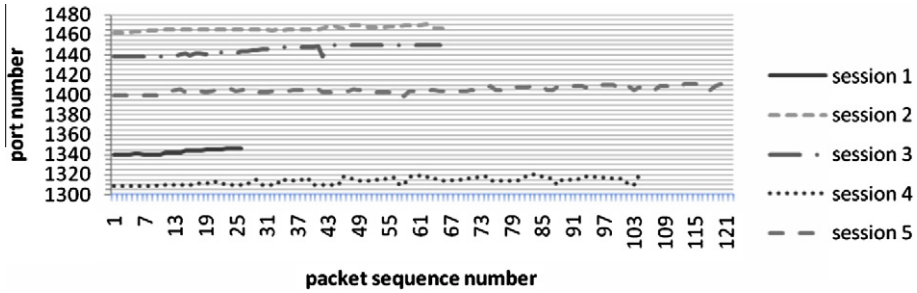


Fig. 4. The port numbers used along with packets during HTTP sessions.

presented because they are all the well-known port, port 80. During each session, the client initiated several flows to communication with the server and the adjacent port numbers used was in an increasing order.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

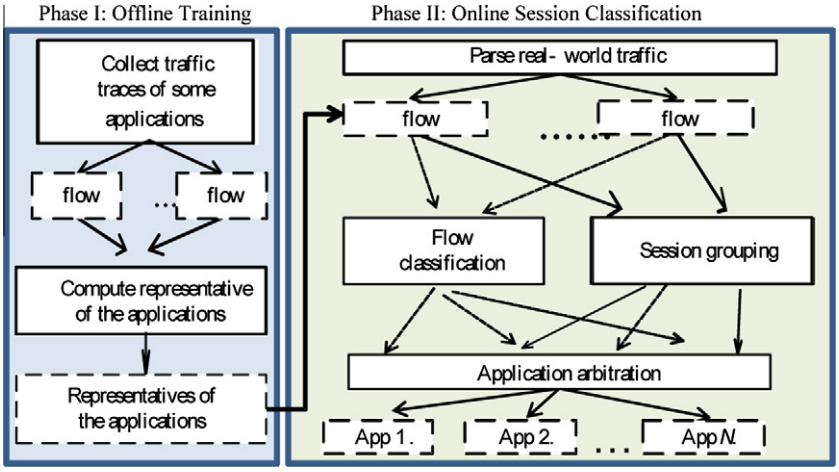


Fig. 5. Components of SLFC.

4. The SLFC algorithm

SLFC runs in two phases: *an offline application representatives training phase* and *an online session classification phase*. Fig. 5 shows the overview of the proposed approach. The left side block represents the steps of the training phase and the right side block shows the online classifier, which includes three modules, *flow classification*, *session grouping*, and *application arbitration*.

The goal of the offline training phase is to find out application representatives, which should be unique to or different from other applications, to be the basis of comparison. Hence, this training phase first collects a set of traffic traces and tries to extract the representatives from the traces. There are two ways that can be used to collect application traffic: (1) capture all traffic generated while some application is executing, and manually filter out the part of traffic unrelated to the application; (2) only capture the part of traffic related to the application. The more pure application traffic is collected, the more accurate the classification results can be expected to obtain because more elaborate application profiles can be reserved. For the second method, a traffic filter can be used to assist this traffic collection process, which can automatically prohibit or filter out irrelevant traffic based on default or common configurations of concerned applications. In terms of concerned applications, the application traffic can be captured with the help of a traffic filter. A traffic filter, like firewall, can be set up to only allow the traffic to pass through some pre-defined ports according to the default or common configurations; in terms of new applications, the configurations of the first occurrence can be saved and a traffic filter can use them to extract and refine the application traffic after filtering out traffic patterns of other known applications.

In order to remain excellent classification, the application representatives should be also kept precise and up-to-date. The cost of representative inspection/upgrade process is acceptable because an automatic approach like the second method as mentioned above can be invoked to compute other new representatives of unknown applications or applications that are revised frequently.

The online session classification phase first extracts the five-tuple information (source IP, source port, destination IP, destination port, protocol) and the packet size distribution from all real-world flows. The packet size distribution of a flow is transformed to a two-dimension space point. Next, the flow classification module compares the flows with application representatives and classifies it into the application having a minimum distance. Afterward, the session grouping module tries to group flows as a session based on port locality. After the above online phases, each flow is classified as an application and port-adjacent flows are grouped into the same session. If two or more flows of a session are classified as different applications, the application arbitration module is invoked to solve the conflict and make the correction.

Each module of SLFC as mentioned above is elaborated in subsequent subsections. Sections 4.1 and 4.2 explain the details of offline training phase and the online session classification phase is interpreted through sections 4.3,4.4,4.5. Section 4.1 describes how a series of packet sizes of flows are converted into representations. A similarity distance metric and four methods that are used to develop the application representatives are described in section 4.2. Along with the application representatives, sections 4.3 and 4.4 introduce the module of flow classification and session grouping respectively. Application arbitration module, described in section 4.5, is used to handle that case if flows within a session are classified as different applications.

4.1. Flow representation – dominating sizes (DS) and dominating sizes' proportion (DSP)

When input into SLFC, successive IP packets having the same 5-tuple are collected as a flow. In order to determine whether any two flows are similar or not, SLFC counts the packets' sizes of each flow. If two flows are similar, the packets' sizes of the two compared flows should be similar. However, exhaustively remembering all packets' sizes of a

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

flow not only consumes a lot of memory spaces but also is impracticable. To overcome this difficulty, only the dominating packets' sizes of a flow are kept as the feature of the flow. Identifying dominating packets' sizes is done as follows. Assume a number of packets are collected for a flow f. First, packets with payload sizes equal to zero or MTU are treated as invalid packets and hence omitted. Second, the number of valid packets for each distinct packet size is counted and stored as a pair of $(ps_f(i), pro(ps_f(i)))$, where $ps_f(i)$ is the ith distinct packet size used by flow f and $pro(ps_f(i))$ is the ratio of $ps_f(i)$ over the total number of valid packets of flow f. Then, all pairs $(ps_f(i), pro(ps_f(i)))$ are sorted in a decreasing order sorting by $pro(ps_f(i))$ and split into two vectors, namely DS_f and DSP_f, which represent the *dominating size* (*DS*) vector and the *dominating size proportion* (*DSP*) vector for flow f, respectively. For example, if a flow f contains packets of h distinct packet sizes, after the sort operation, we have a set of h pairs $\{(ps_f(1), pro(ps_f(1))), (ps_f(2), pro(ps_f(2))), \ldots, (ps_f(h), pro(ps_f(h)))\}$, and the *DS* and the *DSP* for the flow f is denoted as vectors, where $DS_f = \langle ps_f(1), ps_f(2), ps_f(3), \ldots, ps_f(h)\rangle$ and $DSP_f = \langle pro(ps_f(1)), pro(ps_f(2)), pro(ps_f(3)), \ldots, pro(ps_f(h))\rangle$. For the ease of discussion, we use $DS_{f(g)}$ and $DSP_{f(g)}$ to indicate the gth entries in the DS_f and DSP_f vector, respectively. The obtained DS and DSP vectors are also called the PSD feature of a flow.

4.2. Application representatives

In order to measure the degree of similarity for two different flows, a distance metric is defined. However, special handling must be taken when the lengths of *DS* vectors are not the same. Suppose that there are two distinct flows f_1, f_2, and the numbers of entries in *DS* vectors of f_1 and f_2 are n and m respectively ($n \geqslant m$). The similarity distance metric is defined as

$$\text{Similarity distance} = \sum_{g=1}^{m} \sqrt{\left(DS_{f_1(g)} - DS_{f_2(g)}\right)^2 + \left(DSP_{f_1(g)} - DSP_{f_2(g)}\right)^2} + \sum_{g=m+1}^{n} \sqrt{\left(DS_{f_1(g)}\right)^2 + \left(DSP_{f_1(g)}\right)^2}. \quad (1)$$

Here the values of the entries in *DS* vectors are not further normalized because each entry in *DSP* vector is paired with the corresponding one entry in *DS* vector.

The application representatives training phase aims to find out the representatives of pre-selected applications. For this purpose, lots of traces for a specific application are collected and the PSD features of flows of an application are extracted. Below there are four different methods to compute the representatives for an application. One common routine used by the four methods are introduced first. The routine, named representative averaging (RA) algorithm, is used to derive the representative feature from a group of flows. Once flows are correctly grouped, the RA algorithm averages all PSD features of flows which contribute to the corresponding feature values within the group G, i.e. $Rep_G = \{\sum(DS_{f_i}/K), \sum(DSP_{f_i}/K)\}$ for $1 \leqslant i \leqslant K$, where DS_{f_i} and DSP_{f_i} are the *DS* vector and *DSP* vector of flow f_i which contributes to the corresponding feature values within the group G, and K is the total number of flows f_i in group G.

(1) *Method 1* (*M1*) – *direct average processing*: All flows belong to one application contribute to the final representatives of the application. Therefore, all involved flows are used to compute the representative by using the RA algorithm. The result of this method is a single representative for the application.
(2) *Method 2* (*M2*) – *manual traffic correlation:* An application may have multiple different kinds of behaviors. To precisely catch the behavior profiles of an application, this method employs a manual pre-processing stage to classify flows by behaviors. For example, all eMule flows can be classified manually into three behaviors, i.e., connecting to pre-configured servers to fetch server and file lists, communicating with peers, and downloading files. In this case, the representative of the eMule application is composed of three representatives; each is obtained by applying RA algorithms for flows belong to an individual group. Therefore, with *M2*, an application representative may be composed of multiple group-representatives depending on the number of the application behaviors.
(3) *Method 3* (*M3*) – *ignoring common packets*: The method is basically the same as *M1*. However, to prevent the ambiguity brought by packet size similarities, a preprocessing step is done to filter out common packet sizes of different network applications. There exists a tradeoff between the numbers of common packet sizes needed to filter out and the final classification accuracy rates. The more common packet sizes are removed, the higher the separate classification accuracy rate for each application may be achieved. However, the common packet sizes filtered out ultimately can not be recognized. For example, eMule and Skype both use some common size of packets, such as 46 bytes UDP packets, to communicate with peers. The representative finally generated by *M3* is also a single one for each application.
(4) *Method 4* (*M4*) – *automatic clustering*: This method is similar to *M2*, i.e., generate behavior based application representatives. Instead of grouping flows of similar behaviors, it tries to group flows automatically. The basic idea of automatic behavior classification is simple. We assume that flows of the same behavior should have similar PSD features. Hence, a *tolerant threshold (TT)* is defined to tell whether two flows should be grouped together or not. If the PSD distance between two flows is less than *TT*, they are grouped together. Otherwise, they are classified into two different groups. After classifying all flows into behavior groups, the RA algorithm is applied to each group and obtains the corresponding group-representatives. The final representative for the application is composed of all group-representatives.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

Using different application representatives may cause totally different classification results. Readers should also note that the proposed method may generate many representatives for an application because an application can have various implementations and run on different platforms.

4.3. Flow classification

Each incoming flow computes the individual similarity distance between it and all sets of the application representatives found by offline training phase according to the metric Eq. (1). If one certain application has more than one representative, the final distance between the flow and the application is the sum of all similarity distances between the flow and each representative. After all similarity distances are obtained, the incoming flow decides the application having the minimum similarity distance to be the one it should belong to.

4.4. Session grouping

To assist and speed up session identification, a data structure, namely *port association table* (*PAT*), is used to store the port locality information. Once a flow is recognized as a session of a specific application, its four-tuple information (source IP, source port, destination IP, destination port) is extracted and recorded in the *PAT* as (source IP, source port, session ID) and (destination IP, destination port, session ID), where session ID is a counter starting from zero. For a given flow, if its source IP address *SrcIP* is already stored in the *PAT* and the source port number *Q* is adjacent to the port number *P* of an existing (*SrcIP, P, SID*) entry in the PAT, the flow is also treated as a flow of session *SID* and added into the *PAT* as (*srcIP, Q, SID*). The same rule can be applied to destination IP addresses.

However, not all flows with adjacent port numbers actually belong to the same session. For example, two instances of a P2P file-sharing application run at different time, e.g., 3AM and 7PM and some involved flows belonging to different instances use adjacent port numbers. In this case, all involved flows and hosts should be classified into two sessions. Therefore, two parameters, *port locality range* and *flow inter-arrival time*, are defined. Port locality range is used to express numerically the meaning of port locality and indicates if the difference of two port numbers used by two flows is within a specific range, these two flows are classified as the same session; flow inter-arrival time means the difference of separate arrival time of any two flows.

Although port locality is considered, it is not primarily used for application classification; it should be regarded as finding partnership among tremendous flows. The leading part used to classify applications is to compare the traffic characteristics of unknown flows with those of application representatives flow by flow. The port association is to provide the chance to verify the classification results by taking a peek at partners' results. Applications without port locality can be still classified against application representatives, although our approach is unable to find brother flows by 5-tuple flow information in this case.

4.5. Application arbitration

When multiple flows are grouped as a session, it is possible that flows within a session may be classified as different applications. This is because the application classification and flow grouping is done independently. A *classification conflict* happens if a grouped session contains flows of two or more different applications. Although an application may use several communication protocols in a session, flows with different protocols should be classified as the same application. SLFC uses a simple solution to resolve these classification conflicts. If flows of two or more different applications are grouped together, all flows of the session will be treated as the application having the largest amount of flows in this session.

5. Evaluations

In this evaluation, two different data sets are used, both of which are captured and collected from the operational instances of all pre-selected applications running in NCTU campus, not from a traffic generator or artificial design in a lab. For one data set used for training, it contains all pre-selected application traffic and it is only used to compute application representatives. For the other data set, it is only used for the purpose of application identification and classification and is independently captured from the first data set.

For each application, a number of traces are collected and each trace has one or more TCP and UDP flows. Table 2 shows the profile of each pre-selected application. Then the real-world traces recorded from NCTU campus networks are classified to evaluate the proposed solution. Section 5.1 assesses important parameters of SLFC; one is the final representative training method used, one is the tolerant threshold needed for *M4*, and the other two used to delimit a session. The following subsection represents the flow-level and session-level classification accuracy rates. Apart from accuracy rates, section 5.3 shows the critical false positive ratio and false negative ratio. The comparisons with other related proposals are given in sections 5.4 and 5.5. Finally, section 5.6 evaluates the effects of online classification speedup.

5.1. Parameters

In the offline training phase, as described in Section 4.2, we use four distinct methods to extract application representatives. Fig. 6 shows the accuracy rates of four applications based on UDP flow classifications. In *M1*, the accuracy rate of BT and Skype is 0.03% and 0%, respectively, so they are both invisible in the figure. The accuracy rate of *M1* is not good because *M1* may put two flows into a group even if they are far from each other. Averaging different flows that are far from each other may derive a representative which is also far from all flows. *M2* has better accuracy rates than *M1*. Although *M2* has similar results to *M4*, *M2*

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

Table 2
Summarized information of collected application traces.

Application	TCP flows	UDP flows	TCP packets	UDP packets	Total sessions
BT	3365	737820	146844	811609	55
eMule	15255	205545	692642	207195	55
PPLive	8567	12974	9432	12978649	55
Skype	894	6692	12054	40793	55
Shoutcast	40	0	93607	0	55
HTTP	452	0	10759	0	55
POP3	80	0	13215	0	45
SMTP	80	0	10321	0	45
FTP	120	0	25468	0	45

requires a manual traffic pre-processing to obtain *a priori* knowledge about the target application, which can be done automatically in *M4*. *M3* has better accuracy rates than *M1*, but *M3* also requires a pre-processing operation to find out common packet sizes. *M4* has the best accuracy rates among the four methods, so it is chosen as our default method of application representatives. Besides, the two parameters, *port locality range* and *flow inter-arrival time*, used to define a session are set to 4 and 500 seconds according to our observations and the work [14].

The parameter *tolerant threshold* (*TT*) required by *M4* affects the number of flow groups and the accuracy rates of application classifications. Fig. 7 shows the accuracy rates of different values of *TT*. The horizontal axis is the TT value to be evaluated and the vertical one is the accuracy rates of traffic classification. The accuracy rates of $TT = 2$ are similar to that of $TT = 1$ but better than $TT = 5$ and $TT = 10$. Although the results of $TT = 2$ and $TT = 1$ are similar, the numbers of groups are different. Taking PPLive as an example, the number of groups for $TT = 2$ are only two-third times of that for $TT = 1$. Therefore, the value of *TT* is set to two throughout this work.

5.2. Accuracy rate of classifications

In this subsection, two kinds of classification results are shown; one is flow classification accuracy rate and the other is session classification accuracy rate. The accuracy rate of flow classification indicates the percentage of correctly recognized application flows. Two strategies are compared in this evaluation. One is flow-level flow classification that online classifies flows using only PSD features (Flow-level) and the other is session-level flow classification that online classifies flows using whole online classification phase of SLFC (Session-level). Fig. 8 shows the flow classification accuracy rates of the two strategies.

For TCP applications, BitTorrent and eMule have lower accuracy rates than others, because the two applications have similar PSDs, especially when transferring files. For UDP applications, the accuracy rate of Skype is low because Skype has similar packet sizes to eMule and BitTorrent. Compared with P2P protocols, traditional protocols like FTP, POP3, and SMTP have better accuracy rates because they have distinguishable PSDs. Some applications have similar accuracy rates regardless of the use of flow grouping and application arbitration. There are two major reasons for the phenomenon. First, those applications usually use only a single flow to communicate. Second, the port numbers assigned to flows of those applications are not continuous or dynamically assigned. From this evaluation, the session-level flow classification has a high classification accuracy rate, says 99.9% on average and outperforms flow-level flow classification.

In order to find out how many commodity operating systems support to assign port number linearly, we evaluated and found that most current familiar operating systems support this feature if the applications tend to request multiple flows in a short time in an application session. The operating systems, including MS windows XP, windows Vista, windows 7, Ubuntu 8.04, Ubuntu 10.04, Ubuntu 10.10, FreeBSD 8.1, CentOS 5.5 and Fedora 14, all support this feature. The reason is perhaps for quick responses to the requests for resource allocation carried out by operating systems.

Accuracy rates of session recognition represent the percentage of sessions that are correctly classified. It measures

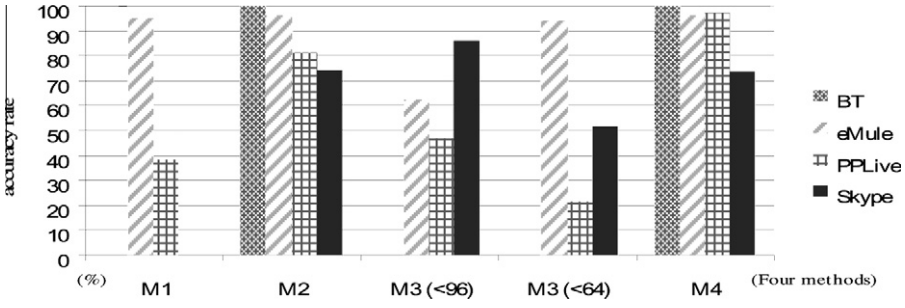


Fig. 6. Four representatives training methods based on UDP flows.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

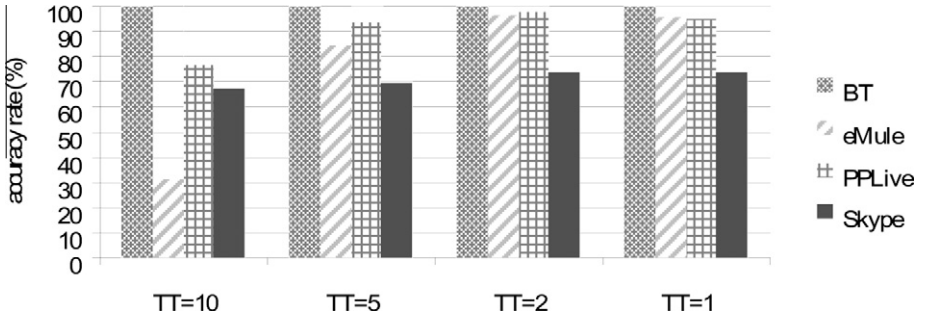


Fig. 7. Tolerant threshold (*TT*) based on UDP flows.

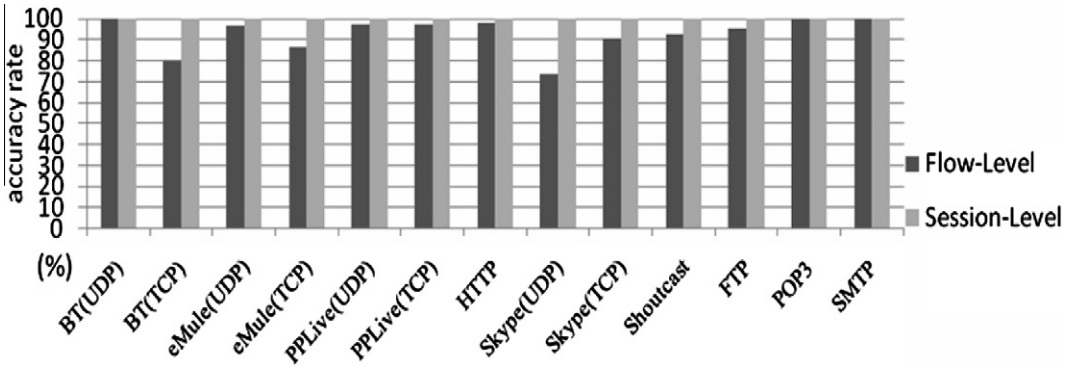


Fig. 8. Accuracy rate of flow classifications.

the accuracy rates of session recognitions by dividing the number of correctly identified sessions by the total number of sessions. Fig. 9 shows the recognition accuracy rate for each TCP application. The horizontal axis lists the applications to be evaluated and the vertical one is the accuracy rates of traffic classification. In most cases, the accuracy rates can reach 100% except the HTTP protocol. This is because connections of an HTTP session can have scattered source port numbers due to redirect requests. If the port numbers are not adjacent and exceed the *port locality range*, these connections will be treated as different sessions, which is an incorrect classification. From this evaluation, a high session classification accuracy rate, says 99.98%, can be obtained.

5.3. False-positive ratio and false-negative ratio

A false positive means that a flow is classified into an incorrect application, while a false negative means a known application cannot be identified. Incorrect classifications are caused by two main reasons. First, if the representatives of two different applications are very similar, an incorrect decision can be made. Second, if the packets collected in one flow are not enough, an erroneous decision can be made as well. For the former one, appropriate representatives that are not similar should be picked up to increase the performance of traffic classification. For the latter one, the classifier is able to collect more packets for a flow to obtain more (*DS*,*DSP*) pairs and hence improve the classification accuracy. False negative may happen because some applications have very diverse behaviors, which are far from those used to training the application representatives.

Table 3 presents false-positive and false-negative rates of each application. False-positive rates are further divided into two cases, i.e., the false-positive rates of using only PSD-based flow classification and the rates of using flow grouping and application arbitration algorithms. Taking BT (TCP) and eMule (TCP) as examples, our method can correctly classify most flows having short-term file transferring, but for long-lasting file transferring, our method may make incorrect decisions because of similar PSDs. In addition, the result of false-positive rates of Skype (UDP) is significant because Skype (UDP) uses similar or same packet size as eMule (UDP). When flows are grouped as sessions, the false-positive rates can be reduced to zero because incorrect classifications can be fixed by application arbitration.

5.4. Compared with K-means

K-means clustering is a method of cluster analysis which aims to partition *n* objects into *k* clusters in which each observation belongs to the cluster with the nearest

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

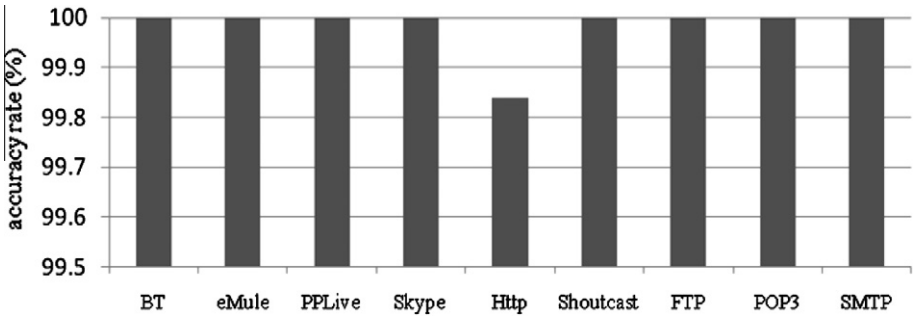


Fig. 9. Accuracy rate of session recognition.

Table 3
False positive and false negative rates of each application.

Application	FP(Flow-level) (%)	FP(Session-level) (%)	FN(Flow-level) (%)
BT (UDP)	0.03	0	0
BT (TCP)	20	0	0
eMule (UDP)	3.57	0	0
eMule (TCP)	13.64	0	0
PPLive (UDP)	2.48	0	0
PPLive (TCP)	2.35	0	0
HTTP	1.67	0	0
Skype (UDP)	26.35	0	0
Skype (TCP)	9.09	0	0
Shoutcast	7.14	0	0
FTP	5	0	0
POP3	0	0	0
SMTP	0	0	0

mean. We compare *M4*, our best application representative training method, with *K*-means and show the flow-level classification accuracy rates of these two methods in Fig. 10. In the figure, the number of groups generated by *M*4 for BT, eMule, Skype, and PPLive are 9, 12, 11, and 21, respectively. Compare with the cases of $K = 9$, 12, 11, and 21, SLFC has a much better classification rates than *K*-means. The results of adopting *K*-means are not as good as *M4* because of two impacts; one is that some flows in one group are far from others and thus derive a representative that is not similar to those flows, and the other is due to our flow representation expressed in the form of *DS* and *DSP* vectors. Besides, the accuracy rate of Skype is lower than others because it has common packet size with eMule and the classification result only benefits either side between eMule and Skype.

5.5. Compared with other related proposals

In this subsection, our method is compared with another two similar researches [11,12], which also classify network flows based on statistical characteristics. The comparison focuses only on five different application protocols, which were evaluated in the two researches. The results are shown in Fig. 11. In [11], the authors adopted two clustering techniques based on vector quantization (VQ) and gaussian mixture models (GMM). The two techniques are labeled as VQ and GMM respectively in the figure.

After analyzing the traffic and classification results, we found that 5-packets approach tried to capture the first five packets shown up in the flow, but did not take the relationship among these packets into consideration. Packets may arrive or be dropped randomly in a congested network. If the number of capturing the original five packets is not enough (less than five) or these packets are out of sequence, 5-packets approach is easily confused and hence usually makes wrong decisions.

Our method has a better accuracy rate because our method can retain all different behaviors of an application by multiple distinct representatives while most other solutions only catch one or parts of behaviors of an application.

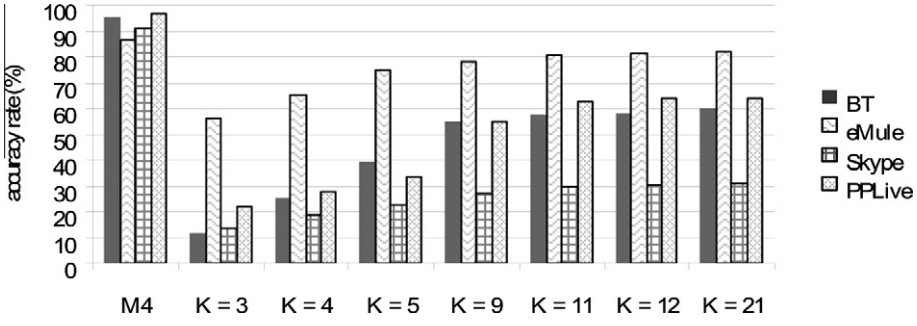


Fig. 10. Compared with *K*-means based on UDP flows.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

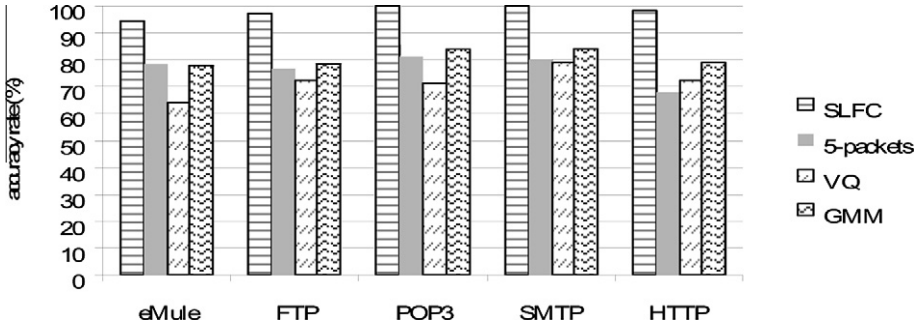


Fig. 11. Compared the proposed solution with 5-packets, VQ, and GMM.

Table 4
Online classification results.

Packet counts	BT (%)	eMule (%)	PPLive (%)	Skype (%)
(a) UDP online classification results				
<10	99.86	94.04	51.7	63.47
<20	99.97	96.43	51.7	63.47
<30	99.97	96.43	72.6	69.46
<40	99.97	96.43	86.3	69.46
Final	99.97	96.43	97.52	72.45

Packet counts	BT (%)	eMule (%)	PPLive (%)	Skype (%)	Shoutcast (%)	FTP (%)	POP3(%)	SMTP (%)
(b) TCP progress results of classification								
<10	21.69	25.8	84.5	68.18	85.71	0	0	0
<50	38.55	34.35	84.5	68.18	85.71	0	0	0
<100	60.24	51.37	84.5	72.73	85.71	18.75	6.25	11.25
<150	72.29	68.39	85.92	72.73	85.71	27.5	6.25	11.25
<200	77.1	85.72	86.38	77.27	92.86	43.75	52.5	57.5
<250	77.1	85.72	88.26	81.82	92.86	72.5	62.5	77.5
<300	79.5	85.72	97.65	81.82	92.86	83.75	81.25	88.75
Final	80	85.72	97.65	90.91	92.86	95	100	100

Application	Total TCP packets	TCP packets inspected	Total UDP packets	UDP packet inspected	Total reduction (%)
(c) Classification cost reduction					
BT	148644	108327	811609	771028	8.42
eMule	692642	486256	207195	196835	24.09
PPLive	9432	8478	12978649	2589240	79.99
Skype	12054	10836	49763	30258	33.52
Shoutcast	93607	13059	0	0	86.04
HTTP	10759	8798	0	0	18.23
POP3	13215	10812	0	0	18.18
SMTP	13021	11328	0	0	13
FTP	25468	10184	0	0	60.01
Total (counts)	1018842	601258	14047216	3587361	–
Total (percentage)	100 %	59.01 %	100 %	25.54 %	72.19

5.6. Classification speedup assessment

In this section, the accuracy rates of online application classifications and the overall classification throughput are evaluated. Table 4(a) and (b) show the intermediate classification results, which are made before a complete flow is seen. The traffic of eMule, PPLive, and Skype may be incorrectly classified because flows of the three applications have some common packet sizes.

From Table 4(b), a decision for a TCP application can be made by checking at most 300 packets and a high accuracy rate can be achieved. In addition, the online throughput of the proposed solution is able to run at a throughput exceeding 400Mbps on a mainstream computer (Intel Core 2 Duo processor 3.0 GHz, 2 GB RAM, and Windows XP operating system). If we checks at most 300 packets for each flow, the number of inspected packets can be greatly reduced, as shown in Table 4(c). An average reduction of 72% can be observed in the table. For TCP and UDP applications, 41.99% and 74.46% packets can be omitted respectively, In particular, for streaming applications such as PPLive (UDP) and shoutcast (TCP), a surprisingly huge reduction of 79.99% and 86.05% are observed respectively.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

6. Conclusions

This paper proposes SLFC, which runs in two phases: an offline application representatives training phase and an online session classification phase. The offline training phase uses a set of traffic traces to find two-dimensional representatives, which are built from PSDs of pre-selected applications. The online session classification is done in three steps: (1) extract the PSD feature of a given flow and compare it with those PSD features of application representatives to make the classification; (2) run the flow grouping algorithm to group port-adjacent network flows into the same session; and (3) use the application arbitration algorithm to correct flow classifications if two or more flows of a session are classified as different applications. SLFC are able to make traffic classifications without examining packet payloads. With a well-trained application representatives' database, the classifier can effectively identify the application that a flow belongs to. The proposed solution achieves high accuracy rates and low error rates. When the proposed solution is used as an on-line classifier, decisions can be made by checking at most 300 packets for long-lasting flows and running at a throughput exceeding 400 Mbps on commodity hardware. Based on out test data, an average of 72% of packets can be omitted without reducing the classification accuracy rates.

References

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, C. Diot, Packet-level traffic measurements from the sprint IP backbone in IEEE Network, IEEE Network (2003). Available from: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1248656&tag=1>.
[2] T. Karagiannis, A. Broido, N. Brownlee, K.C. Claffy, M. Faloutsos, Is P2P dying or just hiding? in: IEEE GLOBECOM, November 2004.
[3] T. Karagiannis, A. Broido, M. Faloutsos, K.C. Claffy, K.C. Transport layer Identification of P2P traffic, in: Internet Measurement Conference(IMC), October 2004.
[4] S. Sen, O. Spatscheck, and D. Wang, Accurate, scalable in network identification of P2P traffic using application signatures, in: WWW2004, May 2004.
[5] M. Roesch, SNORT: Lightweight intrusion detection for networks, in: LISA '99: Proceedings of the 13th USENIX Conference on Systems Administration, November 1999.
[6] V. Paxson, Bro: a system for detecting network intruders in real-time, Computer Networks (1999).
[7] V. Paxson. Empirically derived analytic models of wide-area TCP connections. in: IEEE/ACM Transactions on Networking, August 1994.
[8] V. Paxson, S. Floyd. Wide area traffic: the failure of Poisson modeling, in: IEEE/ACM Transactions on Networking, June 1995.
[9] C. Dewes, A. Wichmann, A. Feldmann, An analysis of internet chat systems, in: IMC, 2003.
[10] M. Roughan, S. Sen, O. Spatscheck, N. Duffield, Class-of-service mapping for QoS: a Statistical signature-based approach to IP traffic classification, in: IMC, 2004.
[11] D.M. Divakaran, H.A. Murthy, T.A. Gonsalves, Traffic modeling and classification using packet train length and packet train size, in: IPOM , 2006.
[12] L. Bernaille, R. Teixeira, I. Akodjenou, A. Soule, K. Salamatian, Traffic classification on the fly, in: ACM SIGCOMM Computer Communication Review, 2006.
[13] Ying-Dar Lin, Chun-Nan Lu, Yuan-Cheng Lai, Wei-Hao Peng, Po-Ching Lin, Application classification using packet size distribution port association, Journal of Network and Computer Applications (2009).
[14] Jayanthkumar Kannan, Jaeyeon Jung, Vern Paxson, Can Emre Koksal, Semi-automated discovery of application session structure, in: IMC, 2006.
[15] T. Karagiannis, K. Papagiannaki, M. Faloutsos, BLINC: multilevel traffic classification in the dark, in: Proceedings of the conference on applications, technologies, architectures, and protocols for computer communications SIGCOMM, 2005.
[16] J. Frank, Machine learning and intrusion detection: current and future directions, in: Proceedings of the National 17th Computer Security Conference, 1994.
[17] A. McGregor, M. Hall, P. Lorier, J. Brunskill, Flow clustering using machine learning techniques, in: PAM, 2004.
[18] S. Zander, T. Nguyen, G. Armitage, Automated traffic classification and application identification using machine learning, in: Proceedings of IEEE Conference on Local Computer Networks (LCN). 2005.
[19] N. Williams, S. Zander, G. Armitrage, A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification, in: ACM SIGCOMM Computer Communication Review, October 2006.
[20] A. Moore, D. Zuev, Internet traffic classification using Bayesian analysis techniques, in: Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETERICS) 2005.

Chun-Nan Lu received his B.S. and M.S. degrees in Computer Science from National Tsing Hua University in 2000 and 2002. He is a Ph.D. student in Computer Science at National Chiao-Tung University. His researches focus on network security and traffic measurement/analysis. He can be reached at cnlu@cs.nctu.edu.tw.

Dr. Chun-Ying Huang received his B.S. in Computer Science from National Taiwan Ocean University in 2000 and M.S. in Computer Information Science from National Chiao-Tung University in 2002. He received his Ph.D. in Electrical Engineering Department from National Taiwan University in 2007. From August 2008, he joined the Computer Science and Engineering Department at National Taiwan Ocean University as an assistant professor. His researches focus on computer network and network security areas, including traffic measurement and analysis, peer-to-peer networking, high performance network algorithm design and implementation, botnet detection, and secure multicast. Dr. Huang is a member of IEEE and ACM.

Ying-Dar Lin received the Bachelor's degree in Computer Science and Information Engineering from National Taiwan University in 1988, and the M.S. and Ph.D. degrees in Computer Science from the University of California, Los Angeles (UCLA) in 1990 and 1993, respectively. He joined the faculty of the Department of Computer Science at National Chiao Tung University (NCTU) in August 1993 and is Professor since 1999. He spent his sabbatical year as a visiting scholar at Cisco Systems in San Jose during 2007–2008. He was the founding director of Institute of Network Engineering during 2005–2007. He is also the founder and director of Network Benchmarking Lab (NBL), co-hosted by Industrial Technology Research Institute (ITRI) and NCTU since 2002, which reviews the functionality, performance, conformance, interoperability, and stability of networking products ranging from switch, router, WLAN, to security, and VoIP. In 2002, he co-founded L7 Networks Inc. which addresses the content networking markets with the technologies of deep packet inspection. At NCTU, he currently directs Computer and Network Center (2007∼), NBL (2002∼),

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007

and D-Link NCTU Joint Lab (2007∼). His research interests include design, analysis, implementation and benchmarking of network protocols and algorithms, quality of services, network security, deep packet inspection, embedded hardware software co-design, wireless mesh, and P2P networking. He is currently on the editorial board of IEEE Communications Magazine, IEEE Communications Surveys and Tutorials, IEEE Communications Letters, Computer Communications, and Computer Networks. He has authored a textbook on Computer Network Experiments. Dr. Lin is a senior member of IEEE. He can be reached at ydlin@cs.nctu.edu.tw and http://www.cs.nctu.edu.tw/∼ydlin.

Yuan-Cheng Lai received the Ph.D. degree in computer science from National Chiao Tung University in 1997. He joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in 2001 and has been a professor since 2008. His research interests include wireless networks, network performance evaluation, network security, and content networking. He can be reached at laiyc@cs.ntust.edu.tw.

Please cite this article in press as: C.-N. Lu et al., Session level flow classification by packet size distribution and session grouping, Comput. Netw. (2011), doi:10.1016/j.comnet.2011.09.007