2012 26th International Conference on Advanced Information Networking and Applications Workshops

Session Level Flow Classification by Packet Size Distribution and Session Grouping

Chun-Nan Lu
Department of Computer Science,
National Chiao-Tung University,
Hsinchu, Taiwan
cnlu@cs.nctu.edu.tw

Ying-Dar Lin
Department of Computer Science,
National Chiao-Tung University,
Hsinchu, Taiwan
ydlin@cs.nctu.edu.tw

Chun-Ying Huang
Department of Computer Science and Engineering,
National Taiwan Ocean University,
Keelung, Taiwan
chuang@ntou.edu.tw

Yuan-Cheng Lai
Department of Information Management,
National Taiwan University of Science and Technology,
Taipei, Taiwan
laiyc@cs.ntust.edu.tw

***Abstract*—Classifying traffic into specific network applications is essential for application-aware network management and it becomes more challenging because modern applications obscure 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 group 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 flow session classifications, say 99.9%. When SLFC is applied to online classification, an average of 72% of packets in long-lasting flows can be skipped without reducing the classification accuracy rates.**

Keywords-flow classification; session grouping; session classification; packet size distribution

I. 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 applications. However, traditional classification methods may not work well for emerging applications because they usually rely on either port number [1, 2] or payload signatures [3-5]. To bypass policies enforced by network monitors, modern applications use several different techniques to cloak their network traffic. Common communication protocols, like HTTP, are often used as covert channels to relay other types of traffic. Both payload 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*. When the PSD of one flow is determined, it is compared to each representative of all pre-selected applications to decide which application it should be. Meanwhile, if the source and destination IP addresses of two flows are the same and their port numbers are adjacent, the two flows may belong to the same session of an application. If flows of a session are classified as different applications, an arbitration mechanism based on majority votes is invoked to make the correction. Evaluations and online benchmarks show

*: This work was supported in part by NSC 100-2221-E-019-045.

978-0-7695-4652-0/12 $26.00 © 2012 IEEE
DOI 10.1109/WAINA.2012.145

that SLFC is able to obtain accurate results and to make decisions by checking at most 300 packets within a flow and the overall throughput exceeds 400 Mbps in a mainstream computer.

This paper is a shortened version of the article [6] that provided more detailed discussion about false-positive and false-negative analysis and comparisons with other related works. In the next section, some important related literatures are surveyed. Section III describes two basic observations, which are the base of designing our classification algorithm. Our SLFC algorithm is formally presented in Section IV and its performance is evaluated in Section V. Finally, conclusions are given in Section VI.

II. 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 statistical features commonly used, for example, contain flow duration, packet inter-arrival time, separate packet size, bytes transferred, number of packets, and etc. However, earlier work only focused on the peculiarities of network traffic classes or applications [7, 8]. Hereafter, more works [9-13] 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.

BLINC, proposed by Karagiannis et al. [14], introduces another type of classification approach 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.

Some alternative proposals [15] utilize machine learning (ML) techniques to network traffic, which are known as a collection of powerful techniques for knowledge discovery and data mining domains. They first use similar statistical features, like aforementioned works, to build models but then apply particular ML techniques, dissimilar with aforementioned works, to classify network traffic. The idea of applying ML techniques for traffic classification was introduced in [16].

III. Features Utilized by SLFC

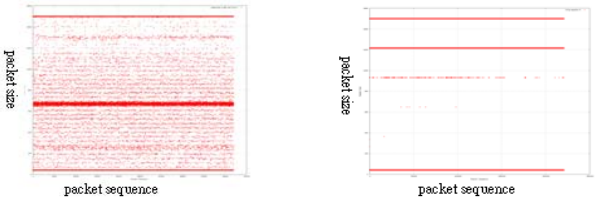
In this section, the two major features utilized by SLFC, i.e., *packet size distribution* (PSD) and *port locality* are introduced. Our observations show that application behaviors can be differentiated with their PSDs, meaning that flows of the same application have similar PSDs, but flows of different 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.

A. Packet size distribution (PSD)

The PSD of a network application can be obtained from all its flows. The traces of individual application activities are captured in a controlled environment. The major advantage of this method is that all collected traffic can be clearly marked to belong to its parent application. Each pre-selected application is executed in turn, and the traffic generated is recorded when it passes through the network interface. The names of the pre-selected applications and related application/protocol category used in this work are BitTorrent (P2P), eMule (P2P), Skype (P2P), HTTP (HTTP), POP3 (POP3), SMTP (SMTP), FTP (FTP), Shoutcast (Streaming) and PPLive (P2P streaming).

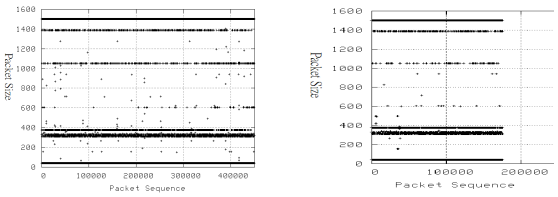
Different applications produce unequal frequent packet sizes due to different operational requirements. Fig. 1(a) and 1(b) show the use of packet sizes of two different applications, Shoutcast and FTP, respectively. The horizontal axis is the packet sequence number and the vertical one is the corresponding packet size. In addition, the packets of the same application have similar size distributions, as shown in Fig. 2(a) and 2(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.



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

Figure 1: Different types of applications have distinct PSD



(a) BitTorrent instance 1 (b) BitTorrent instance 2

Figure 2: Two instances of BitTorrent having similar PSD

B. 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, underlying operating systems often allocate consecutive port numbers when an application has to setup multiple 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 related flows into the same session. However, not all operating systems follow the common rule and hence it is not always able to associate flows as a session. In this case, a single flow is treated as a session.

IV. THE SLFC ALGORITHM

SLFC runs in two phases: an *offline application representatives training phase* and an *online session classification phase*. Fig. 3 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. 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 application traffic. 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. The goals of these two methods are both to clearly mark the related traffic.

In order to remain excellent classification, the application representatives should be also kept precise and up-to-date. The cost of representative 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 header information (source IP, source port, destination IP, destination port, protocol) and the packet size distribution from all incoming flows. The packet size distribution of a flow is transformed to a two-dimension space point. Afterward, the *flow classification* module compares the flows with application representatives and classifies it into the application having a minimum distance. Meanwhile, the *session grouping* module tries to group flows as a session based on port locality. Until now, each flow is classified as a certain 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.

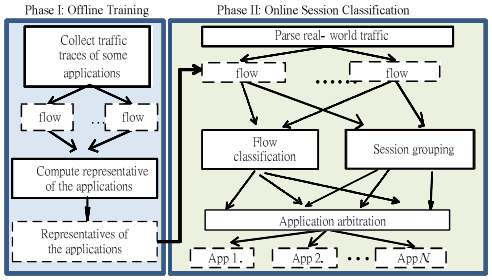


Figure 3: Overview of SLFC

Each module of SLFC as mentioned above is elaborated in subsequent subsections. Subsection A and B explain the details of the offline training phase and the online session classification phase is interpreted through subsection C to E.

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

When input into SLFC, successive IP packets having the same five-tuple are collected as a flow. However, exhaustively remembering all packets' sizes of a 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 a 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. Next, 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 *i*th distinct packet size used and $pro(ps_f(i))$ is the ratio of $ps_f(i)$ over the total number of valid packets. 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 {($ps_f(1)$, $pro(ps_f(1))$), ($ps_f(2)$, $pro(ps_f(2))$), …, ($ps_f(h)$, $pro(ps_f(h))$)}, and the *DS* and the *DSP* for the flow *f* is denoted as vectors, where DS_f =< $ps_f(1)$, $ps_f(2)$, …, $ps_f(h)$> and DSP_f =< $pro(ps_f(1))$, $pro(ps_f(2))$, …, $pro(ps_f(h))$>. For the ease of discussion, we use $DS_{f(g)}$ and $DSP_{f(g)}$ to indicate the *g*th 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.

B. Application representatives

In order to measure the degree of similarity for two different flows, a distance metric is defined. However, careful 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 ($n \geq m$). The *similarity distance* (SD) metric is defined as

$$SD = \sum_{g=1}^{m} \sqrt{(DS_{f_1(g)} - DS_{f_2(g)})^2 + (DSP_{f_1(g)} - DSP_{f_2(g)})^2} + \sum_{g=m+1}^{n} \sqrt{(DS_{f_1(g)})^2 + (DSP_{f_1(g)})^2} \quad (1)$$

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 then 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), 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. $\mathrm{Rep}_G = \{\sum(DS_{f_i}/K), \sum(DSP_{f_i}/K)\}$ for $1 \leqq i \leqq K$, 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) Method1 (*M1*) – *Direct Average Processing*: All flows belong to one application are used to compute the representative by using the RA algorithm. The result of this method is a single representative for the application.

2) Method2 (*M2*) – *Manual Traffic Correction*: 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 calssified manually into three behaviors, i.e., connecting to pre-configured servers to fetch server and file lists, communicating with peers, and exchanging 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) Method3 (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) Method4 (*M4*) – *Automatic Clustering*: This method is similar to *M2*, i.e., generate behavior-based application representatives. Instead of grouping flows of similar behaviors manually, it tries to group flows automatically. We observed that flows of the same behavior should have similar PSD features. Hence, a *tolerant threshold* (*TT*) is defined to tell whether or not two flows should be grouped together. If the PSD distance between two flows is less than *TT*, they are grouped together. Otherwise, they are classified into two different groups. Afterward, 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.

Using different application representatives may cause totally different application 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, and they can choose the most suitable representatives against those results.

C. Flow classification

Each incoming flow computes the individual similarity distance between it and all sets of the application representatives found by the offline training phase according to the metric Equation (1). If an 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.

D. 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 five-tuple header information is extracted and separately 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 entry (*srcIP*, *P*, *SID*) 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 also 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 pre-defined 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 five-tuple flow information in this case.

E. 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 flow classification module and session grouping module do their tasks 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.

V. 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.

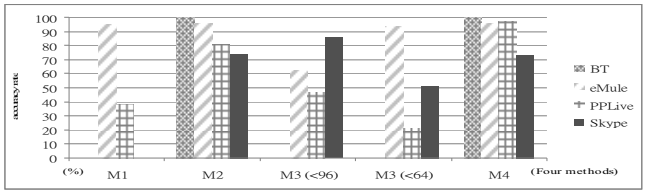


Figure 4: Four representatives training methods based on UDP flows

For each application, a number of traces are collected and each trace has one or more TCP and UDP flows. Then the real-world traces recorded from NCTU campus networks are classified to evaluate SLFC. The following subsection A assesses important parameters of SLFC; one is the final representative training method used, another is the tolerant threshold needed for *M4*, and the other two used to delimit a session. Subsection B represents the flow-level and session-level classification accuracy rates, and subsection C evaluates the effects of online classification speedup.

A. Parameters

In the offline training phase, four distinct methods are designed to extract application representatives. Fig. 4 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 this 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* 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 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 [15].

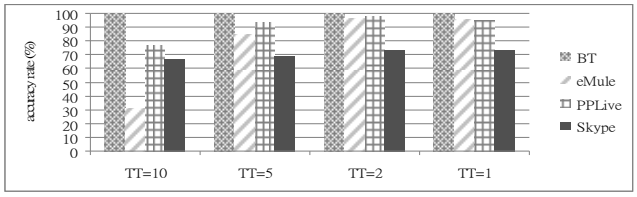


Figure 5: *Tolerant Threshold* (*TT*) based on UDP flows

The parameter *tolerant threshold* (*TT*) required by *M4* affects the number of flow groups and the accuracy rates of application classifications. Fig. 5 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 different values of *TT*. 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.

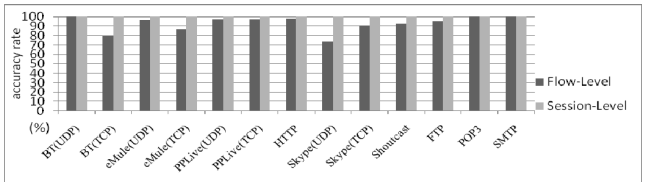


Figure 6: Accuracy Rate of Flow Classifications

B. Accuracy rates of classifications

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. 6 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 session 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.

C. Classification speedup assessment

In this section, the accuracy rates of online application classifications and the overall classification throughput are evaluated. Table 1(a) shows 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 1(a), 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 400 Mbps on a mainstream computer (Intel Core 2 Duo processor 3.0 GHz, 2 GB RAM, and Windows XP). If we checks at most 300 packets for each flow, the number of inspected packets can be greatly reduced, as shown in table 1(b). An average reduction of 72% can be observed in the table.

Table 1(a): TCP progress results of classification

Packet counts	BT (%)	eMule (%)	PPLive (%)	Skype (%)	Shoutcast (%)	FTP (%)	POP3 (%)	SMTP (%)
< 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

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.

Table 1(b): Classification Cost Reduction

Application	Total TCP packets	TCP packets inspected	Total UDP packets	UDP packet inspected	Total 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	—
Percentage	100 %	59.01 %	100 %	25.54 %	72.19

VI. 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 application representatives, which are built from PSDs of pre-selected applications. The online session classification phase is done in three steps: (1) extract the PSD feature of a give flow and compare it with those PSD features of application representatives to make the classification; (2) run the session 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 online 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 our 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*, November 2003.

[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*, Nov. 1999.

[6] C.N. Lu, C.Y. Huang, Y.D. Lin, Y.C. Lai, "Session Level Flow Classification by Packet Size Distribution and Session Grouping," *Computer Networks*, In Press, 2011.

[7] V. Paxson. "Empirically derived analytic models of wide-area TCP connections." in *IEEE/ACM Transactions on Networking,* Aug 1994.

[8] V. Paxson and S. Floyd. "Wide area traffic: the failure of Poisson modeling." in *IEEE/ACM Transactions on Networking*, June 1995.

[9] C. Dewes, A. Wichmann, and A. Feldmann, "An analysis of internet chat systems," in *IMC* 2003.

[10] M. Roughan, S. Sen, O. Spatscheck, and 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] Y.D. Lin, C.N. Lu, Y.C. Lai, W.H. Peng, P.C. Lin, "Application classification using packet size distribution and port association," in *Journal of Network and Computer Applications*, March 2009.

[14] Karagiannis T, Papagiannaki K, Faloutsos M., "BLINC: multilevel traffic classification in the dark," in *ACM SIGCOMM CCR*, Oct. 2005.

[15] T.T. Nguyen, G. Armitage, "A survey of techniques for Internet traffic classification using machine learning", in *IEEE Communications Surveys and Tutorials, 2008*.

[16] J. Frank, "Machine learning and intrusion detection: current and future directions," in *Proceedings of the National 17th Computer Security Conference*, 1994.