Unknown Pattern Extraction for Statistical Network Protocol Identification

Yu Wang, Chao Chen and Yang Xiang
School of Information Technology, Deakin University
Melbourne, Australia
{y.wang, zvm, yang.xiang}@deakin.edu.au

***Abstract*—The past decade has seen a lot of research on statistics-based network protocol identification using machine learning techniques. Prior studies have shown promising results in terms of high accuracy and fast classification speed. However, most works have embodied an implicit assumption that all protocols are known in advance and presented in the training data, which is unrealistic since real-world networks constantly witness emerging traffic patterns as well as unknown protocols in the wild. In this paper, we revisit the problem by proposing a learning scheme with unknown pattern extraction for statistical protocol identification. The scheme is designed with a more realistic setting, where the training dataset contains labeled samples from a limited number of protocols, and the goal is to tell these known protocols apart from each other and from potential unknown ones. Preliminary results derived from real-world traffic are presented to show the effectiveness of the scheme.**

Keywords-network protocol; machine learning; semi-supervised learning; constrained clustering

I. INTRODUCTION

Most advanced network management and security functions require the ability of identifying network protocols in real-time and dividing network traffic into separate queues accordingly. The classic techniques, such as port-based filtering and payload-based deep packet inspection (DPI), have shown their limitations in the face of the rapidly growing Internet and evolving network applications. For example, a lot of P2P applications evade the detections by using dynamic port negotiation and encryption [1].

In response, a lot of research efforts have been dedicated to statistics-based protocol identification. In this approach, some of the discriminative flow-level and packet-level characteristics of traffic flows (e.g. statistics of packet size and inter-packet arrival time) are extracted, so that the flows are represented in the form of feature vectors, which can provide input to machine learning (ML) algorithms. Generally, the learning engines take in a set of labeled flow samples for training and then return a classification model that can predict the types of protocols for future flows. A number of classic ML algorithms have been investigated in this regard [2-10], and the empirical results suggest that accurate and fast protocol identification can be achieved.

Nonetheless, some important aspects of this approach remain unexplored. In specific, most prior works formulate the problem as a supervised multi-class classification problem, which implies the fundamental assumption of complete a priori knowledge. In other words, it is assumed that all existing protocols are known in advance and sufficient samples of each protocol are presented in the training set. Obviously, this is not the case in real-world deployment scenarios. First of all, collecting and labeling a large amount of training samples for every existing network protocol is extremely expensive if not impossible. At the same time, new applications and protocols keep emerging and patterns of known protocols also shift over time. Facing the unknown protocols and patterns that are never seen during training, the classifiers trained with the complete a priori knowledge assumption will fail, while such traffic will be falsely classified to an arbitrary known class rather than labeled as novelty or anomaly. This problem has been overlooked in most previous studies, because their evaluations are also designed and carried out in the ideal settings where the training and testing data are homogeneous (i.e., unknown traffic is filtered out from testing set).

In this short paper, we revisit the statistical network protocol identification problem with a much looser assumption on a priori information. Specifically, we assume that only a limited number of protocols are known at the time of training and there are a lot of unknown ones in the wild. It reflects a realistic scenario where the network operators are in possess of two types of data, i.e., labeled data and unlabeled data. Labeled data represent a number of known protocols, which are the targets to identify in real-time traffic monitoring. Unlabeled data can be easily collected in the operating networks and they comprise a mixture of data samples from both the known protocols and unknown ones. Based on this setting, we propose the *Lunex* scheme (**L**earning with **un**known pattern **ex**traction), which learns from both labeled and unlabeled data for the potential capability of detecting unknown protocols. We use real-world Internet backbone traffic data to evaluate the proposed scheme along with various existing learning schemes. The results suggest that the proposed scheme enables unknown protocol detection for statistical network protocol identification.

The remainder of the paper is organized as follows. Section II reviews the existing schemes and section III describes the proposed learning scheme. In section IV, preliminary results are presented and analyzed. Section V concludes the work.

II. RELATED WORK

The past decade has seen a large body of research focused on statistical network protocol identification. Moore and Zuev [2] explore Naive Bayes and its refined variants for categorizing IP traffic. Auld et al. [3] extend their work in [2] by investigating the classifier based on Bayesian neural networks. Williams et al. [4] present a comparison of five supervised learning algorithms with a focus on feature selection and run-time performance. Kim et al. [5] compare statistical traffic classifiers with classic port-

This work is supported by ARC Discovery Project DP150103732 and ARC Linkage Project LP120200266.

based approach and host behaviour-based Blinc [16]. Este et al. [6] analyse the stability of information in statistical flow features with respect to time and location of observation. Lim et al. [7] show that port numbers, lengths of first few packets and feature discretization are critical sources of discriminative power for network protocol identification. Zander et al. [8] explore the issues in practical implementation and deployment through a runtime performance evaluation based on DIFFUSE.

In the above studies, the potential unknown protocols (i.e., those absent from the training data) are overlooked, not only in their system design but also in the evaluation. In contrast, a few works propose training binary or one-class classifiers that have the potential to detect unseen patterns. Crotti et al. [9] propose statistical protocol fingerprints for HTTP, SMTP, and POP3. The fingerprints are one-class classifiers that determine whether a flow is a match by calculating its normalized anomaly score. Este et al. [10] use one-class SVM classifier for general protocol identification. For each protocol in the training data set, they first derive a one-class SVM model from positive samples, and then tune the decision boundary by using negative samples as well as a set of hypothetical outlier data samples derived from a uniform distribution. Nguyen et al. [11] develop binary classifiers for ET (an online game) and VoIP traffic using Naïve Bayes and C4.5 decision tree. They focus on identifying long-lived interactive flows using sub-flow statistics, and their results indicate that the binary classifiers can obtain good precision and recall rate. Also, Erman et al. [12] propose a semi-supervised learning algorithm that performs k-means clustering on a large amount of unlabeled data and a small amount of labeled data. In this work, we extend their idea by performing constrained clustering for the unknown pattern extraction. The above works show that unknown traffic can be detected if the classification schemes are designed more carefully, but the effectiveness of the detection is tested in an ad-hoc way such that the actual performance against the variety of unknown patterns on the Internet is unclear.

In the case where only unlabeled traffic data is given, cluster analysis is a feasible technique to discover patterns within the data in an unsupervised manner. Some classic algorithms have been applied to derive traffic clusters in prior studies, including expectation maximization [13], DBSCAN and K-Means [14, 15]. In [16], we propose a constrained clustering algorithm that improves the purity of traffic clusters to up to 98%.

III. METHODOLOGY

Network protocol identification is the process of associating network traffic flows with the application-layer protocol in use. A traffic flow in this context comprises the packets exchanged between two particular endpoints, which share the same 5-tuple of source IP address, source port number, destination IP address, destination port number, and transport layer protocol.

The statistical approach identifies the underlying protocols of traffic flows based on the observations of some packet and flow level characteristics. That is, each flow is described by a feature vector comprising its values on a fixed, predefined set of features. In this work, we define flow features using some simple statistics (including maximum, minimum, mean, and standard deviation) of packet size and inter-packet arrival time [16]. The features are measured based on the first ten packets in the flows and they are calculated separately in each direction.

TABLE I. CONFUSION MATRIX OF A 6-CLASS CLASSIFICATION MODEL

% classified as →	bt	bttp	http	ssh	https	smtp	other
bt	**97.3**	0.5	1.1	-	0.8	0.3	0
bttp	1.1	**97.5**	1.1	-	0.3	-	0
http	0.2	1.9	**96**	-	1.8	0.1	0
ssh	0.1	-	-	**99.6**	0.2	0.2	0
https	0.6	0.2	3.3	-	**95.4**	0.5	0
smtp	0.1	0.1	0.2	-	0.6	**99.1**	0
ftp	0.6	-	0.4	-	3.5	95.6	**0**
pop3	0.2	-	26.1	-	1.2	72.4	**0**
razor	-	-	-	-	-	100	**0**
imaps	7.4	0.4	11.9	15.7	62.2	2.4	**0**
dns	17.6	-	32.2	2.2	43.1	4.9	**0**
pop3s	0.1	-	29.1	1.9	45.7	23.2	**0**
x224	9	0.4	9.7	4.1	75	1.8	**0**
aim	1.6	0.2	0.3	20.5	0.7	76.7	**0**
policy	-	-	47.4	16.3	-	36.3	**0**
mysql	16.9	0.2	70.8	-	1.6	10.6	**0**
rtmp	1	-	59.3	-	39.7	-	**0**
imap	14.6	-	11.1	4.9	54.2	15.3	**0**
ymsg	34.7	-	1.3	1.3	61.3	1.3	**0**
rsp	100	-	-	-	-	-	**0**
unidentified	5.8	0.3	1.2	91.9	0.6	0.2	**0**

Suppose we are given a set of labeled data consisting of n flows from K protocols: $D = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$, in which each $\boldsymbol{x}_i \in \mathbb{R}, (i = 1, \dots, n)$ is the feature vector of a flow, and $y_i \in \{\omega_1, \dots, \omega_K\}, (i = 1, \dots, n)$ is its corresponding class label. Based on D, we can train a multi-class classification model that maps any input feature vector to one of the output classes, i.e., $F(\boldsymbol{x}): \mathbb{R} \to \{\omega_1, \dots, \omega_K\}$. Alternatively, we can train a series of binary classifiers, i.e., $F_c(\boldsymbol{x}): \mathbb{R} \to \{\omega_c, \overline{\omega_c}\}, (c = 1, \dots, K)$, each of which predicts whether a given flow belongs to ω_c or not.

In real-world deployment, the classification models will face unknown protocols and novel patterns that they have never seen during the training phase. That is, the testing data set consists of l flows from U unknown protocols in addition to the K known protocols: $T = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_l, y_l)\}$, in which the class labels are $y_i \in \{\omega_1, \dots, \omega_K, \varphi_1, \dots, \varphi_U\}, (i = 1, \dots, l)$. However, most existing schemes overlook the unknown protocols such that their traffic will be falsely classified into an arbitrary known protocol. As an example, Table I shows the confusion matrix for a random forest classifier trained with labeled data of six protocols in our data set (see section IV). We can see that the classifier accurately identifies flows of the known protocols, but it fails to tell apart the patterns from unknown protocols.

In this work, we propose the *Lunex* scheme that learns from both a labeled training data set $D = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$ and an unlabeled data set $D' = \{(\boldsymbol{x}_1, ?), \dots, (\boldsymbol{x}_m, ?)\}$. D' is expected to consist of data from the known classes $\omega_1, \dots, \omega_K$ as well as unknown ones $\varphi_1, \dots, \varphi_U$. It then returns a classification model $F(\boldsymbol{x}): \mathbb{R} \to \{\omega_1, \dots, \omega_K, \overline{\omega}\}$, which is able to identify the known patterns and put the unknown ones into a novelty class $\overline{\omega}$.

The *Lunex* scheme works in two stages (see Fig. 1). The first stage is unknown pattern extraction. It takes both data sets as input, and searches in the unlabeled data for novel patterns that are absent from the labeled data. In specific, the mixed data set are firstly partitioned into clusters using a constrained clustering algorithm [16], and then the clusters are labeled according to the majority of labeled data inside them. The unlabeled clusters (i.e., those having no labeled data assigned to them) are then extracted and the data within them are considered as novel patterns.

In the second stage, *Lunex* takes in the labelled data set and the extracted unlabelled clusters, and it returns a series of binary classifiers. In particular, for each of the known protocols ω_c, a binary classification model, i.e., $F_c(\boldsymbol{x})$: $\mathbb{R} \rightarrow \{\omega_c, \overline{\omega_c}\}$, is trained by using its labelled data as positive samples and the rest data (including the labelled data from other known protocols and also the extracted data) as negative samples. The classifier acts like a protocol signature during testing, as it matches whether a traffic flow belongs to the corresponding protocol ω_c or not. Therefore, to classify a testing flow, each model $F_c(\boldsymbol{x}),\ c = 1, \ldots, K$ gives a prediction. If there is one and only one match, the flow will be classified to the matched protocol. If there is no match at all, it will be considered as a novel pattern belonging to an unknown protocol (i.e., class $\overline{\omega}$). If there are more than one match, we will resort to an additional multi-class model for the final decision.

IV. Experimental Results

A. Experimental Setup

The evaluation of the proposed scheme is conducted based on a real-world Internet traffic data set, i.e., the wide traces [16], collected on 30th and 31st of March in 2012. We use a customized DPI tool to build the ground truth manually, and use Wireshark protocol analyzer for validation. For the purpose of evaluation, we focus exclusively on TCP traffic, in which we identify about 20 protocols. In particular, there are 6 dominating protocols that constitute over 50 thousand flows in the data set, i.e., BitTorrent (bt), BitTorrent Tracker (bttp), HTTP, HTTPs, SSH, SMTP. Other protocols include POP3, FTP, RAZOR, DNS over TCP, IMAPs, X224, POP3s, AIM, Macromedia Policy (policy), MySQL, RTMP, IMAP, Yahoo Messenger (ymsg), and RSP. Besides, about 17% of the traffic cannot be identified by our DPI tools. Traffic in the first and the second day is used to form the training and testing data set respectively. Each set consists of up to 20,000 random flows for each identified protocol and 50,000 random flows for the unidentified traffic. In the experiments, we consider 6 dominating protocols as the target (known) protocols to identify, while the other protocols and unidentified traffic are unknown. To form the labeled data set, we randomly select and label 2,000 flows for each target protocol from the first day.

We use overall flow accuracy and three per-class metrics (i.e., f-measure, precision and recall) to measure the performance. Overall accuracy is the percentage of correctly classified flows over the total number of flows. For each particular protocol ω_c, true positives (TP) is the number of its flows that are correctly recognized (i.e., $\omega_c \rightarrow \omega_c$), false negative (FN) is the number of its flows that are incorrectly rejected (i.e., $\omega_c \rightarrow \overline{\omega_c}$), and false positive (FP) is the number of flows of other protocols that are mistakenly accepted (i.e., $\overline{\omega_c} \rightarrow \omega_c$). Accordingly, precision for ω_c is calculated as TP/(TP+FP) and recall is TP/(TP+FN), and then f-measure equals (2×precision×recall)/(precision +recall).

B. Results and Analysis

In the experiments, we compare the Lunex scheme with three existing works, including one-class SVM [10], Semi-supervised learning [12], and traditional multi-class supervised classifier as adopted in [2-8]. For the latter, we test a number of popular ML algorithms and present the best candidate (i.e., Random Forest). Random Forest is also used as the base classifier in the proposed Lunex scheme.

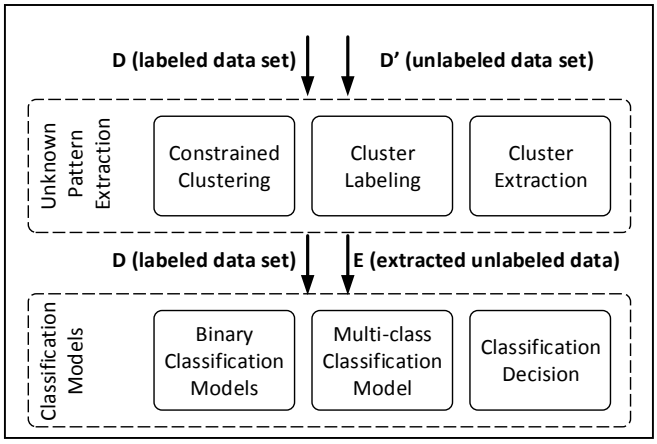

Figure 1. Framework of the Lunex Scheme

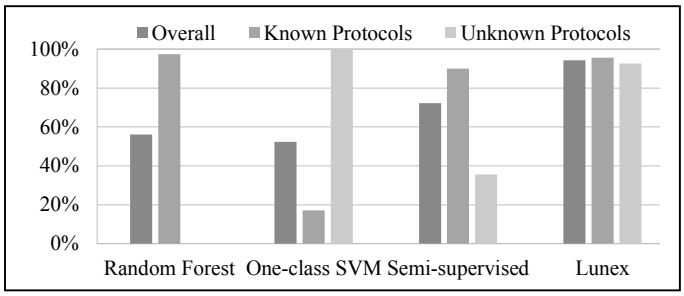

Figure 2. Classification Accuracy Results

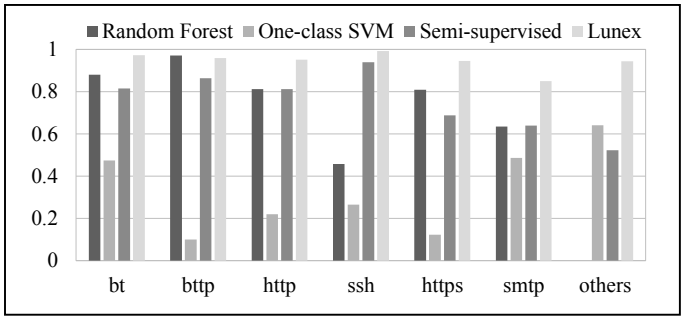

Figure 3. Per-class F-Measure Results

The flow accuracy results are presented in Fig. 2. It can be noticed that only 56% of the testing flows are correctly identified by Random Forest, which is the best multi-class classifier trained with labeled data solely. The accuracies of one-class SVM and semi-supervised learning are 52% and 72% respectively, and the proposed Lunex scheme outperforms the others by successfully identifying 94% of the flows.

Fig.2 also presents classification results for known protocols and unknown protocols separately. First of all, Random Forest identifies 97.5% flows of known protocols but it misclassifies all unknown flows due to the lack of novelty detection mechanism. Second, the decision boundary learned by one-class SVM is too biased towards the positive class, such that it rejects all unknown patterns but accepts only 17% of flows from known protocols. Third, semi-supervised learning recovers 90% of flows for the known protocols and 35.6% of flows from the unknown ones. In comparison, Lunex identifies 96% of flows of known protocols and detects 92.6% of unknown flows.

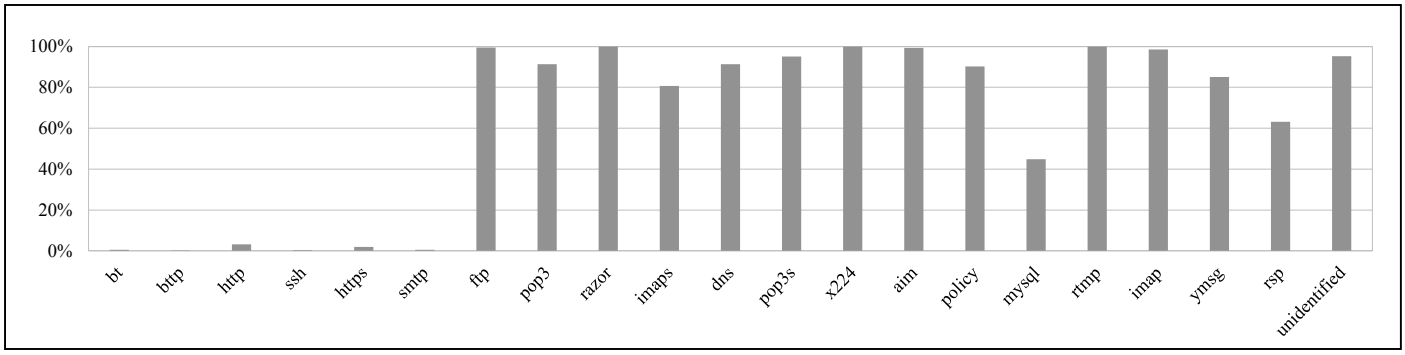

Figure 4. Unknown Extraction Result

Fig. 3 details the per-class f-measure results for known protocols and the general unknown class, in which we can see that Lunex achieves best per-class performance in general. In particular, it derives over 0.95 f-measure for all classes except for smtp (0.84). Take a closer look at smtp, we find that its recall rate is 0.99, but the precision rate is only 0.76 due to relatively high FPs, which are caused by the poor predictions on some pop3 (an unknown protocol) flows. We believe this is because smtp and pop3 are both mailing protocols and they do share some similarity in traffic pattern. Further information and extra features can be extracted in order to distinguish the two more accurately, which is out of the scope of this work.

The above results show that the proposed unknown pattern extraction scheme is able to effectively improve the accuracy of protocol identification by mining the unlabeled data for the missing a priori knowledge for unknown protocols. Next we examine the behavior and performance in the unknown pattern extraction stage. The unlabeled data in our evaluation consist of 6 known protocols and 15 unknown ones, and Fig. 4 reveals the extraction rate for each of these protocols.

First, Lunex successfully extracts over 80% of flows from 13 unknown protocols, while for the rest 2 unknown protocols, i.e., mysql and rsp, the extraction rates are between 40% and 60%. Second, it can be seen that very few unlabeled flows from the 6 known protocols are falsely extracted as novel patterns (which will become noise). In specific, 3% of http flows and 1.9% of https flows are extracted, while the figures for the other known protocols are negligible. In general, Lunex extracts 38.5% of unlabeled flows with a 98.4% true positive rate (note that here true positive means that an extracted flow is indeed unknown).

We also find that by increasing the size of labeled data set we can extract more information about unknown protocols from unlabeled data, and thus the classification accuracy of Lunex can be further improved. The related result is not presented here due to the page limits, but it can be found at http://anss.org.au/nsclab.

V. CONCLUSION

In this short paper, we report the preliminary results of the proposed unknown pattern extraction scheme for statistical network protocol identification. The proposed scheme extends the statistics-based traffic classifiers with the ability to detect novel patterns that are unknown to them, which makes them useful in real-world application scenarios. Experimental result shows that the scheme achieves 94% flow accuracy that breaks down to 96% for known protocols and 92.6% for the unknown.

REFERENCES

[1] T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” IEEE Commun. Surveys Tuts., vol. 10, no. 4, pp. 56-76, FOURTH QUARTER 2008.

[2] A. Moore and D. Zuev, “Internet traffic classification using Bayesian analysis techniques,” in ACM International Conference on Measurement and Modeling of Computer Systems, June 2005.

[3] T. Auld, A. W. Moore, and S. F. Gull, “Bayesian neural networks for Internet traffic classification,” IEEE Trans. Neural Networks, no. 1, pp. 223–239, January 2007.

[4] N. Williams, S. Zander, and G. Armitage, "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification," ACM SIGCOMM CCR, Vol. 36, No. 5, Oct 2006, pp 7-15.

[5] Hyunchul Kim, Dhiman Barman, kc claffy, Michalis Faloutsos, Marina Fomenkov and KiYoung Lee, "Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices", in Procedding of the ACM CoNEXT 2008.

[6] A. Este, F. Gringoli and L. Salgarelli, On the Stability of the Information Carried by Traffic Flow Features at the Packet Level, ACM SIGCOMM Comput. Commun. Rev., 39 (3), pp. 13-18, 2009.

[7] Yeon-sup Lim, Hyun-chul Kim, Jiwoong Jeong, Chong-kwon Kim, Ted “Taekyoung" Kwon and Yanghee Choi, "Internet Traffic Classification Demystified: On the Sources of the Discriminative Power", in Proceeding of the ACM CoNEXT 2010.

[8] S. Zander, G. Armitage, "Practical Machine Learning Based Multimedia Traffic Classification for Distributed QoS Management," in LCN 2011, Bonn, Germany, 4-7 October 2011.

[9] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, “Traffic classification through simple statistical fingerprinting,” ACM SIGCOMM Comput. Commun. Rev., vol. 37, no. 1, pp. 5–16, 2007.

[10] A. Este, F. Gringoli, and L. Salgarelli, "Support Vector Machines for TCP traffic classification", Computer Networks, Vol. 53, No. 14, 2009, pp. 2476-2490.

[11] T. Nguyen and G. Armitage, “Clustering to Assist Supervised Machine Learning for Real-Time IP Traffic Classification,” in proceeding of ICC 2008, pp 5857-5862, 2008.

[12] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson, “Semi-supervised network traffic classification,” SIGMETRICS Performance Evaluation Review, vol. 35, no. 1, pp. 369–370, 2007.

[13] S. Zander, T. Nguyen, and G. Armitage, “Automated traffic classification and application identification using machine learning,” in IEEE LCN '05, Sydney, Australia, November 2005.

[14] J. Erman, M. Arlitt, and A. Mahanti, “Traffic classification using clustering algorithms,” in Proceeding of MineNet ’06, pp. 281–286.

[15] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, “Traffic classification on the fly,” SIGCOMM CCR,vol. 36- 2, 2006.

[16] Yu Wang, Yang Xiang, Jun Zhang, Wanlei Zhou, Guiyi Wei, Laurence T. Yang, "Internet Traffic Classification Using Constrained Clustering", IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 11, pp. 2932 – 2943, Nov 2014.