International Journal of Future Generation Communication and Networking
Vol.6, No.5 (2013), pp.11-20
http://dx.doi.org/10.14257/ijfgcn.2013.6.5.02

NTCA: A High-Performance Network Traffic Classification Architecture

Guanglu Sun, Hui Dong, Dandan Li and Feng Xiao

Research Center of Information Security & Intelligent Technology
School of Computer Science and Technology
Harbin University of Science and Technology
Harbin, China
guanglu.sun@gmail.com, donghui19870714@gmail.com, dandanli@gmail.com

Abstract

Traffic classification is critical to effective network control and management. Recent researches on Internet traffic classifications have developed several methods for identifying types of application, which have advantages in certain types of network traffic. However, these methods are powerless to measure the network traffic with dynamic port, encrypted payloads, mixing traffic, and real-time traffic. In response to the growing requirements of traffic classification for increasingly complex network environment, this paper introduces network traffic classification architecture (NTCA) with high performance. By combining port-based, signature string matching, regular expression matching, and machine learning methods, NTCA achieves high speed and accuracy traffic classification. The experimental results show that our proposed method is able to achieve over 95.0% in average accuracy for all testing traces.

Keywords: *Traffic classification, Architecture, NTCA, High-performance*

1. Introduction

Traffic classification acts as an essential part in common network management such as intrusion detection and network monitoring. Network managers or Internet Service Providers (ISP) can enforce organization policies or offer support to many current and next generation services according to the classification results. Therefore, accurate and fast traffic classification is one of the keystones in network management and monitoring.

The early traffic classification methods are based on port numbers of transport layer, typically registered in Internet Assigned Numbers Authority (IANA). In the recent years, more and more popular applications such as those peer-to-peer (P2P) applications utilize dynamic ports, or utilize other protocols as wrappers, which lead to port-based traffic classification less reliable [1].

Payload-based classification methods deeply inspect the contents in network packets and identify applications by a set of unique payload signatures which represent the different network applications respectively. Signatures and regular expressions are two methods that describe patterns of protocol formats. These approaches achieve high accuracy of classification and are widely used [2]. However, the associated high computational cost, especially in regular expression matching, as well as the increasingly application of encrypted payloads, makes payload-based methods face performance bottlenecks in real world networks [3].

Neither the port-based methods nor the payload-based methods can analyze the traffic with encrypted and unknown protocols effectively, which are continuously increasing and occupy the main bandwidth in the network, such as P2P and Stream. The host-behavior- based methods identify these types of traffic in order to ensure the formal applications in the network. BLINC is a typical host-behavior-based method, which analyzes the behaviors of the transformation between different hosts [4]. However, it analyzes the traffic in off-line mode, so that it does not fit the real-time traffic classification.

In recent 10 years, a lot of researches focus on machine learning (ML) methods with flow-level features for traffic classification [5, 6]. ML methods can achieve over 90% accuracy in the experimental environment. However, ML methods rely on gold standard training datasets and suitable flow-level features, which is still a big challenge in real-time network environment.

In order to meet the growing requirements of traffic classification for increasingly complex network, this paper introduces high-performance architecture for network traffic classification, NTCA. The above effective methods including port-based, payload-based, flow features and ML-based method are merged in terms of NTCA implementation. These methods are hung in NTCA as modules. According to different demands of traffic classification, different methods are selected and combined like a procession.

The remainder of this paper is organized as follows. Section 2 describes the framework of NTCA. In Section 3, our classification methodology is formally stated. We evaluate the accuracy and performance of our proposed methods in Section 4. Finally, the conclusion and some future work are given.

2. The Framework of NTCA

NTCA is mainly used to analyze the application protocol of each flow in the network in order to effectively control and manage the network bandwidth and quality of service. NTCA is approximately a real-time classification method which can support the retransmission and linkage. The main function of NTCA contains: (1) Traffic acquirement and storage; (2) Traffic processing and constructing the flow table; (3) Traffic classification; (4) Result statistics and display.

Throughout the paper, we define flows according to their 5-tuple (source and destination IP address, source and destination port, protocol) and have it expire in 64 seconds [7]. We process network traffic with CAIDA’s CoralReef suite [8].

The framework of NTCA is shown in Figure 1. First of all, the traffic pre-processing module analyzes packets captured from real-time network. NTCA processes packets in 2-4 network layers. Because almost all the application protocols are based on TCP/IP protocol stack, NTCA focus on the packets with Ethernet, IPv4(6), TCP and UDP protocol. After that, the tuple5 of packets are utilized to build the hash table, which is mainly used for storing the flow-level features, payload of packets and interaction information. The information packet length and flow duration are relatively less in data volume, so they are easily stored. However, the payload of packets is too larger to be stored in condition of large-scale real-time network traffic. To tackle this problem, we only store 1500 bytes or 7 packets for each flow.

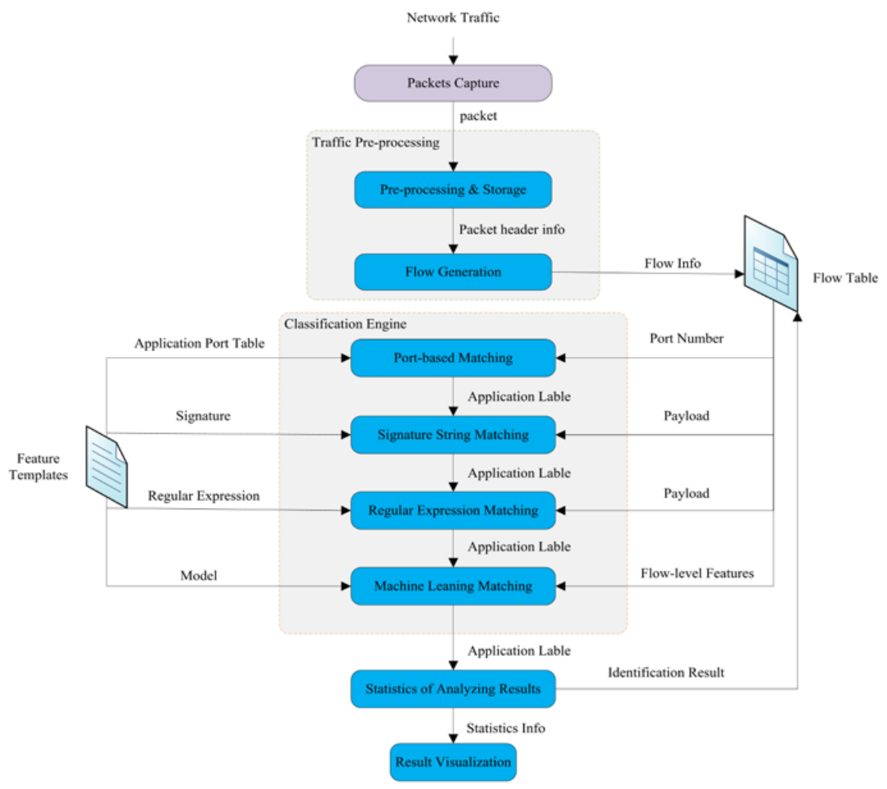

Figure 1. The Framework of NTCA

The information stored in the flow table can be used to analyze the application protocol of the flow in classification engine. The engine consists of four main modules, port-based matching module, signature string matching module, regular expression matching module and machine leaning module. The feature templates provide application port table, protocols signatures, regular expressions and machine learning model for classification engine. Each classification module can not only be utilized independently, but combined with each other to analyze complicated protocols, as shown in algorithm 1. The results are written back to the flow table, when the flow is identified successfully.

Algorithm 1:

```
input : flow f
$l \leftarrow application(f);$
if $l = Unknown$ then
    $l \leftarrow result(Port - based\ module);$
end
if $((l = Unknown) \| (l = HTTP))\ \&\&\ (packet(l) \leq 7)$ then
    $l \leftarrow result(Signature - based\ module);$
end
if $((l = Unknown) \| (l = HTTP))\ \&\&\ (payLength(l) \leq 1500)$ then
    $l \leftarrow result(Regex - based\ module);$
end
if $(l = Unknown) \| (l = HTTP)$ then
    $l \leftarrow result(ML\ module);$
end
$application(f) \leftarrow l;$
```

The results of the flow table are written to the database at regular time. Meanwhile, global information including the number of packets, IP packets, TCP packets, UDP packets, is written to the database. The visualization module displays the statistical results stored in the database with various types of diagrams and data format.

3. Traffic Classification Engine

The main traffic classification modules in NTCA are port-based matching, signature string matching, regular expression matching and machine leaning method.

3.1. Port-Based Matching

Port-based matching module uses the important port number to decide the corresponding application name in application port table (APT) except HTTP. There are two steps in this module:

A. Important Port Selection

The important port is the port number that is necessary for traffic classification in port-based matching. Generally, most clients do not use port number less than 1024 as random port numbers; the randomly system-generated port numbers are always larger than 1024. However, the port numbers belong to servers are always less than 1024 and not changed. We select the important port in a flow by examining port numbers of

Table 1. An Example of the Application Port Table

Application Name	TCP Well-known Ports	UDP Well-known Ports
FTP	20,21	
MSN messenger	1863, 6981-6990, 14594	
Emule	4662	4672
eDonkey	4661,4662,6667	
Microsoft Media Services	1755,2948,2949	1755
Games	3427, 6112, 6881	6112

less than 1024. If there are no port numbers less than 1024 in a TCP connection, we analyze the 3-way handshaking in the TCP connection sequence. We look for SYN or SYN-ACK packets during session establishment to find the server side of a new client-server TCP connection, and then the port of server is selected as important port. For UDP connections, we consider the destination port number of the first packet in one flow as the important port.

B. Application Port Table Construction

Application port table contains the corresponding relation between the port numbers and the applications. We determine the application name, port number and transport layer protocol of these applications. Though many popular applications use dynamic ports, some of them always use several certain port numbers (*e.g.*, eDonkey uses 4661 and MSN messenger uses 1863), those usually larger than 1024. We get these corresponding relations by off-line statistics and previous studies [9], and extend them to the IANA port list, a small portion of APT is shown in Table 1.

Copyright ⓒ 2013 SERSC

3.2. Signature String Matching

Signature string matching module inspects payloads for specific signature of known applications. If the signature of one application in feature templates is matched, then marks it with this application. However, it costs a lot of time because of matching each packet. In our method, we develop two optimization methods to improve signature string matching's efficiency. On one hand, signature_mask is proposed to improve the processing speed. In the initialization of the signature string matching, exclusive disjunction "||" is used to divide the signature of protocols into three groups, which are TCP_payload, TCP_no_payload and UDP. Through this division, an input flow is only matched with the signatures of the protocol group that it belongs to. On the other hand, signature string matching module only processes payload volume less than 1500 bytes or 7 packets, because most flows can be classified within only first 7 packets.

We use the signature string matching method developed in [10], and augmented with more payload signature from OpenDPI [11]. Signature string matching module can identify more than 100 popular applications.

3.3. Regular Expression Matching

Traditionally, payload-based methods inspect packets payload by comparing messages within packets' payload to a set of strings, which represent signatures from a given application. Recent years, some researches focus on a combination of heuristics and regular expression matching, which brings better accuracy.

Following the famous protocols identification tool L7-filter, regular expression matching module in our method consists of the description files for protocols' format using regular expression, Non-Deterministic Finite Automata (NFA) transmission module, the matching module with PCRE. The matching module uses multi-payload matching mode. That is, in the limitation of accumulative payload length, the matching module connects the new payload to the existing payload and matches the joint payload with NFA when the new packet arrives. The matching module gives up the identification until the accumulative payload length exceeds the limitation. The empirical value of limitation is 1500 bytes. We can identify over 50 types of applications with the regular expression matching module.

3.4. Machine Learning Method

To achieve high accuracy of classification in machine learning model, we should consider two main levels, they are that:

A. Select a Set of Discriminating Features

One of the pre-processing steps of machine learning is feature selection. We must select a subset of flow features to acquire higher learning accuracy with lower computational complexity. With the IP packets restructuring and TCP packets restructuring algorithm in Libnids, bidirectional flow features of both TCP and UDP packets are selected as well as backbone network and edge network traffic. 87 bidirectional flow features are applied, most of which were inspired from the 248 bidirectional flow features used in [12].

Considering the classification accuracy and efficiency, we choose the Correlation-based Filter (CFS) as flow features selector [13]. The Correlation-based Filter calculates the relevance between flow features and selected features, which are highly correlated to specific application but with minimal correlated to each other.

B. Select an Effective Classification Algorithm

Considering a sample set $X = \{x_1, x_2, \ldots, x_n\}$ belongs to a application set $C = \{c_1, c_2, \ldots, c_n\}$. The feature set belongs to an application set $C = \{c_1, c_2, \ldots, c_n\}$. The feature set belongs to a sample x_i is defined as $F_i = \{f_{i1}, f_{i2}, \ldots, f_{im}\}$. Those features are numeric or discrete values. For a spending sample x^*, machine learning methods compute the conditional probability $p(c_i|x^*)$ based on different learning models. We use Bayesian methods and Support Vector Machines (SVM) in our experiments, which are computationally practical and outperform the other machine learning methods in terms of classification accuracy and acceptable computational cost.

3.5. The Hybrid Method

The above 4 classification modules have respective advantages. Port-based matching module has the lowest computation cost. Signature string matching module and regex expressions matching module have high accuracy, and machine learning matching module is able to handle encrypted traffic. However, they can hardly satisfy the demands of accurate and rapid when they are applied separately. We develop a hybrid method by combining these methods. A typically scenario of the hybrid method is identifying encrypted traffic those based on SSL protocol.

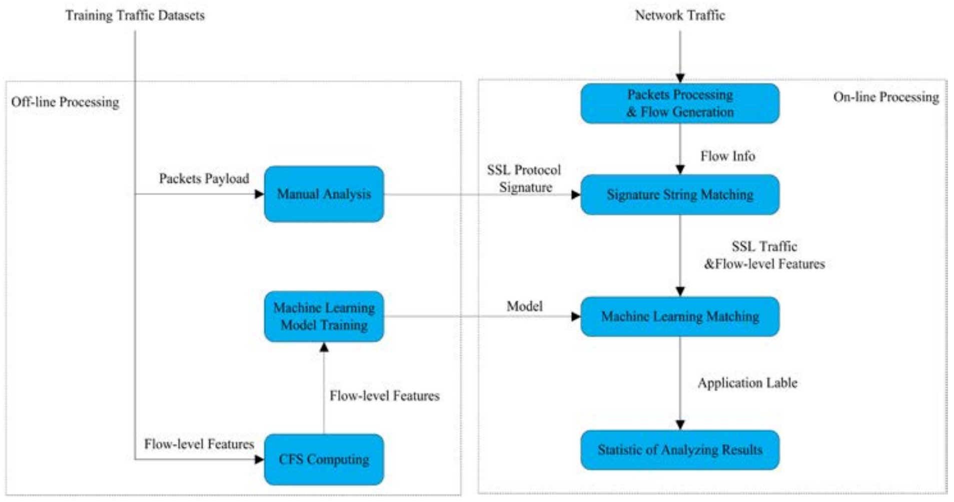
Training Traffic Datasets
Network Traffic
Off-line Processing
On-line Processing
Packets Processing & Flow Generation
Flow Info
Packets Payload
Manual Analysis
SSL Protocol Signature
Signature String Matching
SSL Traffic &Flow-level Features
Machine Learning Model Training
Model
Machine Learning Matching
Application Lable
Flow-level Features
Flow-level Features
CFS Computing
Statistic of Analyzing Results

Figure 2. The Hybrid Method of SSL based Traffic Identification

Figure 2 depicts a typical application of our hybrid method in identifying encrypted traffic based on SSL protocol. Firstly, SSL traffic is separated from the other traffic with the signature string matching module. Second, we identify protocols (*e.g.*, Tor, HTTS, and Windows Update protocols) those based on SSL protocol by machine leaning module with flow features mentioned in Section 3.4.

4. Experiments on Traffic Classification

4.1. Data Sets

We evaluate the proposed methods on six real-world traces from diverse network environments, collected in China (CERNET-1, CERNET-2) and another from a US-Japan

Trans-Pacific backbone link (WIDE). Our edge network traces are collected from a 1 Gb/s Ethernet link in Harbin University of Science and Technology in China (HUST-1, HUST-2). The last data sets collected from the sender of our local area network, mainly contained encrypted flows based on SSL protocol and some Non-SSL flows (AREA).

Table 2 describes the detail of each trace. All the traces contain payload information, thereby enabling us to establish a reference point using the payload-based classifier. The payload-based classifier examines each packet's payload against our array of feature templates, and in case of a match, classifies the corresponding flow with an application label. Previously classified flows are not re-examined again unless they have been classified as HTTP, in which case some applications (*e.g.*, P2P, Streaming) establish their communication sessions with HTTP messages [14]. For the encrypted flows (SSL), we use machine learning methods and manual analysis for further identification, in which case encrypted applications communicate based on SSL protocol. For each trace, we classify flows into 11 groups of applications, namely, Web (WWW), Peer-to-Peer (P2P), Chat, Games, Network Management (Net.oper.), Streaming, Encryption, Mail, DNS, FTP, and Unknown. Figure 3 lists the traces profile used as reference for evaluating our method. Our traces vary widely in applications, which motivating to our evaluation analysis.

Table 2. Characteristics of the Six Data Sets

Data Sets	Date	Link Type	Src.IP	Dest.IP	Packets	Bytes	Flows
CERNET-1	2010/11/15	Backbone	380K	430K	160M	100G	20M
CERNET-2	2010/10/16	Backbone	23M	30M	1260M	800G	190M
WIDE	2006/03/03	Backbone	263K	794K	32M	14G	1872K
HUST-1	2012/12/29	Edge	1270K	1740K	160M	100G	7720K
HUST-1	2012/12/25	Edge	1290K	1770K	180M	110G	8030K
AREA	2010/10/03	Edge	34K	53K	30M	10G	130K

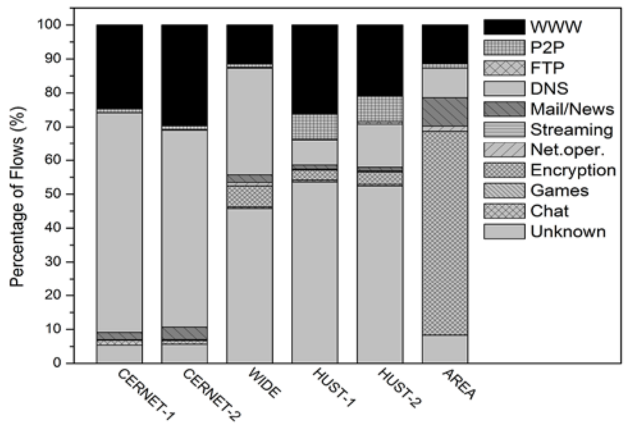

Figure 3. Analyzed Traces Profile

4.2. Performance Metrics

To measure the performance of our method, we apply four metrics, namely, total accuracy, precision, recall, and F-score. Total accuracy is used to measure the accuracy of our method on all traces and the others are used to evaluate the performance in classifying each application. These metrics are defined as following:

- $Total\ accuracy = \dfrac{sum(TP)}{sum(TP)+sum(FP)}$
- $Precision = \dfrac{TP}{TP+FP}$
- $Recall = \dfrac{TP}{TP+FN}$
- $F\text{-}score = \dfrac{(\beta^2+1)\times Precision\times Recall}{\beta^2\times Precision+Recall}$

Where TP represents True Positive, FP represents False Positive, FN represents False Negative, respectively. The value of β is usually defined as 1.

4.3. Experimental Methodology

Our experiments are designed in two aspects. Firstly, the performance of the hybrid method is confirmed by analyzing the classification result of SSL-based traffic. Secondly, we evaluate the overall performance of NTCA and compare with port-based methods, payload-based methods, and machine leaning (ML) methods on the six data sets.

A. SSL-based Traffic Analyzing

The hybrid method is applied to classifying SSL-based encrypted traffic which is difficult to analyze with payload-based matching. We identify SSL traffic according to [15], which can achieve over 99% accuracy. Background flows contain of SSL-based encrypted traffic and Non-SSL traffic collected from the sender of our local area network (AREA). SSL-based flows contain different SSL-based encrypted traffic, namely, Tor, Https, Windows updates, Oscar, Oovoo, Jaber and others.

For our testing, SVM method achieves over 95.0% average accuracy on our traces with 5000 training flows. We use 6 flow-level features in our experiments, including mean packet length, maximum and minimum packet length, mean inter-arrival time of packets, maximum and minimum inter-arrival time of packets, flow duration and the size of the first six packets. Figure 4 shows the performance of the hybrid method in identifying encrypted flows with the independent assumption based on the SVM model.

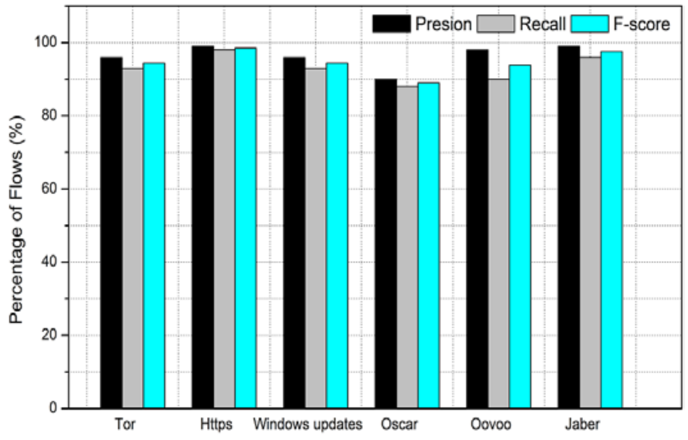

Figure 4. Performance of the Hybrid Method

As shown in Figure 4, Tor, Https, Windows, Updates, Oscar, Oovoo, and Jaber are well classified based on SVM model. We achieve over 90.0% accuracy in classifying each application and 96.0% average accuracy. Moreover, compared with single use of ML methods, the hybrid methods show better performance in real-time classification.

B. The Classification Performance of NTCA

We evaluate the performance of NTCA on the six data sets. In our experiments, previously classified flows are not examined again unless they have been classified as SSL, in which case some applications communicate based on SSL protocol. For all the data sets, NTCA has a high performance on various kind of traffic, and achieves over 95.0% accuracy in Figure 5 (a).

We compare NTCA to port-based methods, payload-based methods and machine learning methods on the same traces. Figure 5 (b) shows the comparison results. On the first five data sets, that is, CERNET-1, CERNET-2, WIDE, HUST-1, HUST-2, payload-based methods can achieve more than 93.0% accuracy. However, payload-based methods are powerless to measure the encrypted network traffic, only achieve 40.0% accuracy in data set AREA. Though machine learning methods achieve over 90% accuracy in all the data sets, their efficiency, real-time capability and accuracy are still fall behind our requirements in real traffic environments.

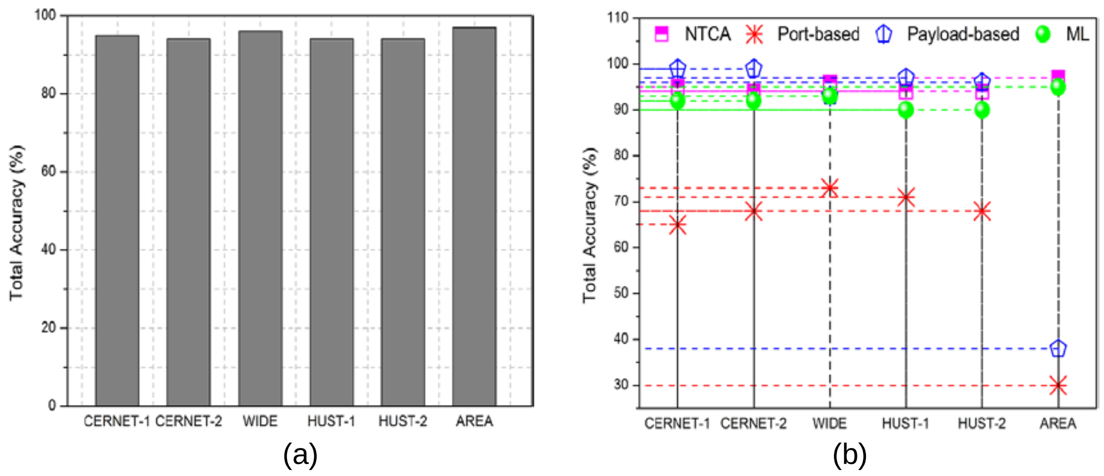

Figure 5. Performance of the NTCA Method: (a) Total Accuracy of NTCA on all the 6 Traces; and (b) Comparing NTCA to Port-based Methods, Payload-based Methods, and Machine Learning Methods on the Same Traces. SVM Model is used in Machine Learning Methods with 5000 Train Flows

5. Conclusions and Future Work

Measuring the mixture network traffic is important for network control and management. This paper proposes new architecture, named NTCA, to classify the network traffic with high performance. NTCA combines port-based methods, signature string matching methods, regex expression matching methods and machine learning methods. It can process the mixture network traffic with more than 95.0% accuracy. Furthermore, it identifies SSL-based encrypted traffic in average accuracy of 96.0%.

We are pursuing this work in three points. The first is focusing on applying the hybrid method to measure P2P protocols. The second is searching more effective flow-level features.

The third is applying our traffic classification architecture to the actual system for the real-time traffic.

Acknowledgments

This research is supported by the graduate students' innovative research projects in Heilongjiang province (YJSCX2012-125HLJ), Research fund for the program of new century excellent talents in Heilongjiang provincial university (1155-ncet-008), China National Nature Science Grant (60903083), and Scientific planning issues of education in Heilongjiang Province(GBC1211062). We thank the anonymous referees for their useful and illuminating comments. Special thanks to Hyunchul Kim and Paul Hick for sharing their codes and data sets.

References

[1] K. P. Moore, “Toward the accurate identification of network applications”, Proceedings of Passive and Active Measurement Workshop, **(2005)** April, pp. 41-54.
[2] S. Sen, O. Spatscheck and D. Wang, “Accurate, scalable in network identification of P2P traffic using application signatures”, Proceedings of the 13th International Conference on World Wide Web, New York, USA, **(2004)** May, pp. 512-521.
[3] S. Fernandes, R. Antonello and T. Lacerda, “Slimming down deep packet inspection systems”, IEEE Conference on Computer Communications Workshops, **(2009)** April, pp. 1-6.
[4] T. Karagiannis, K. Papagiannaki and M. Faloutsos, “Blinc: Multilevel traffic classification in the dark”, ACM SIGCOMM, **(2005)** August, pp. 229-240.
[5] A. Moore and D. Zuev, “Internet traffic classification using Bayesian analysis techniques”, ACM Sigmetrics Performance Evaluation Review, vol. 33, no. 1, **(2005)**, pp. 50-60.
[6] T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning”, IEEE Communications Surveys and Tutorials, vol. 10, no. 1-4, **(2008)**, pp. 56-76.
[7] K. Claffy, H.-W. Braun and G. Polyzos, “A parameterizable methodology for Internet traffic flow profiling”, IEEE Journal on Selected Areas in Communications, vol. 13, no. 8, **(1995)**, pp. 1481-1494.
[8] K. Keys, D. Moore and R. Koga, “The architecture of the CoralReef: Internet traffic monitoring software suite”, Passive and Active Measurement, **(2001)**.
[9] M. Kim, Y. Wonand and J. Hong, “Application-level traffic monitoring and an analysis on IP networks”, Etri Journal, vol. 27, no. 1, **(2005)**, pp. 22-42.
[10] H. Kim, K. claffy and M. Fomenkov, “Internet traffic classification demystified: myths, caveats, and the best practices”, Conference on Emerging Network Experiment and Technology, no. 11, **(2008)** December, pp. 1-12.
[11] http://www.opendpi.org.
[12] A. Moore and D. Zuev, “Discriminators for use in flow-based classification”, Technical Report, Intel Research, Cambridge, **(2005)**.
[13] N. Williams, S. Zander and G. Armitage, “Evaluating machine learning algorithms for automated network application identification”, Technical Report 060401B, CAIA, Swinburne Univ., **(2006)** April.
[14] E. Freire, A. Ziviani and R. Salles, “On metrics to distinguish skype flows from HTTP traffic”, Journal of Network and Systems Management, vol. 17, no. 1-2, **(2009)**, pp. 53-72.
[15] G. Sun and Y. Xue, “An novel hybrid method for effectively classifying encrypted traffic”, IEEE Global Communications Conference, **(2010)** December, pp. 1-5.

Copyright ⓒ 2013 SERSC