An Efficient Method to Maintain the Header Signatures for Internet Traffic Identification

Sung-Ho Yoon, Myung-Sup Kim
Dept. of Computer and Information Science
Korea University
Seojong, Korea
{sungho_yoon, tmskim}@korea.ac.kr

Abstract— **With the rapid development of the Internet in recent years, there has been a growing emphasis on importance of application traffic identification for efficient network management. Although various identification methods have been proposed, it is difficult to apply them to real networks due to several problems. To overcome the limitations of existing methods, we proposed the header signatures, i.e., the 3-tuple {IP address, port number, and L4 protocol (TCP/UDP)} of the packet header that represents either a dedicated or a temporary Internet application (service) server. However, a problem with this method is that the number of signatures increases rapidly because header signatures are composed of combinations of header elements in a rapidly changing network, as many servers providing application lunch and disappear in every day. In this paper, we propose an efficient method for signature maintenance using the weight of the signature, which is measured by a function consisting of maintenance elements.**

Keywords—header signature; network management; traffic classification; traffic identification

I. INTRODUCTION

The application-level traffic identification, that ascertains which application is contributing to the network traffic, has to be performed before we can apply the various network management policies. The final goal of traffic identification methodology and system is to accurately name all the network traffic corresponding to each application. Methods that use diverse traffic features for traffic identification have been suggested, but there are limitations when applying these methods to a real network.

In earlier work, we proposed the use of header signatures as well as management methods to overcome the limits of existing identification methodology [1]. The header signature is the header information of the server that offers the specific application for either an extended or short period of time. Thus, the header signature can overcome the limitations of traffic identification effectively because it identifies network traffic simply by matching the signature and the packet header. Although the header signature can overcome the shortcomings of existing methodologies effectively, it has a problem in that the number of signatures increases steadily. This problem arises due to the characteristics of the header signature, namely, that it is composed of a combination of header information. Not all signatures can be stored because of the limited size of the physical memory.

To resolve this issue, we propose a maintenance function that recalculates the weight of the signature based on statistical information of identified traffic. We also propose a maintenance function that recalculates the weight of the signature at specified time intervals to adopt the network environment changes. Furthermore, we compare the proposed maintenance method and the earlier method with respect to several evaluation metrics.

This paper is organized as follows. We examine the existing traffic identification methods in Chapter 2. The header signature and the identification system structure are explained in Chapter 3. In Chapter 4, we propose our maintenance method, and compare it with existing methods by using the evaluation metrics in Chapter 5. Lastly, we put forward our conclusions and future work in Chapter 6.

II. RELATED WORKS

Several traffic identification methods that use header information have been suggested to supplement the accuracy of port-based analysis method [2][3]. A common assumption in these methods is that the Internet application server provides the same application (or service) continually for a given period of time.

Moore et al. [2] demonstrated the inaccuracy of port-based identification method by means of experiments. In their research, they used the "host/port" of identified traffic to validate specific application traffic such as port scanning and streaming audio. The host/port information is maintained when it is active. Baldi et al. [3] configured a Service Table by extracting "network coordination" from the result of existing method such as payload and used it to analyze traffic. To maintain the service table, elements of the service table are deleted if they are not used for a specified amount of time.

In the aforementioned papers, it is asserted that the header information of the server that provides application can be used effectively for traffic analysis. However, the authors focus on the use of header information, rather than the management of header information. The only exception is Baldi et al. [3] who propose a maintenance method based on inactivity time of the signature. However, this method (in

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (2010-0020728) and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2007483)

which elements are deleted if they are not used for a certain amount time) results in a situation where the header information of applications that have long service periods is deleted frequently. Thus, we require maintenance methods for preserving the signature that have a possibility of being used in future identifications, and for deleting the signatures that will not be required in future.

III. HEADER SIGNATURE

As the traffic behavior becomes more complex, the analysis of overall traffic behavior based on packet units becomes more difficult. To supplement this analysis, a flow unit was proposed. A flow is a set of packets having the same 5-tuple (source IP address, destination IP address, source port, destination port, L4 protocol). Hence, we define F as a set of flows (f), and $F(x)$ as a set of flows having the same property x, where x is any combination of 5-tuple header entries. This is given in (1) as follows:

$$F = \{f_1, f_2, f_3, \dots, f_n\} \quad (1)$$

Although the flow unit is effective in analyzing massive amount of traffic comparing to packet units, it does not provide sufficient information about traffic behavior because it is a set of packets generated from the end points of particular two given hosts. Additional information is required for analyzing the relation between flows in the server–client model. This analysis is carried out by aggregating associated flows.

$$B = \left\{ b | b = \left\{ f | f \in F(x), x = \{IP, Port, Protcol\} \right\} \right\} \quad (2)$$

For effective signature creation, we define the term "Bunch" to reflect the relation between flows. A bunch is a set of flows having the same 3-tuple that is shown in (2). In other words, it represents all flows that connect to a specific server port. A bunch consists of one server node and a set of flows between counter peers (client).

The header signature proposed in this paper is a server node of the bunch. It consists of header information (x), application (a), and the weight (w). The header information is a 3-tuple (IP address, port, L4 protocol) representing the server; the application is a service provided by the server. The weight is a value that is measured using the identified traffic. This value is used for signature maintenance and is measured by the maintenance function $M(x)$.

$$HS = \{(x, a, w) | x = \{IP, port, protocol\}, w = M(x)\} \quad (3)$$

Figure 1 shows the system architecture of the identification system. The system consists of three parts. In the creation part, it converts packets to flows (Flow Generation), and extracts signatures (Signature Extraction). The created signature is stored in the central signature DB. The identification part converts the test network traffic into flows (Flow Generation). It compares the header information with the header signature (Flow Identification). Thus, it names the application to the network traffic if the traffic and the signature have the same header information. In addition, the identification part updates several maintenance elements of the signatures used (Weight Feedback). In the maintenance part, it updates the weight value of each signature by means of a maintenance function that uses maintenance elements (Signature Maintenance), and names application (Signature Naming). We separate the application naming from the creation part. If we name the application in the creation part, the range of creation decreases because the only signatures knowing their application name can be created as signature. Hence, the header information is extracted in the creation part. Following this, the application is named in the maintenance part using various results such as payload and statistic signature identification.

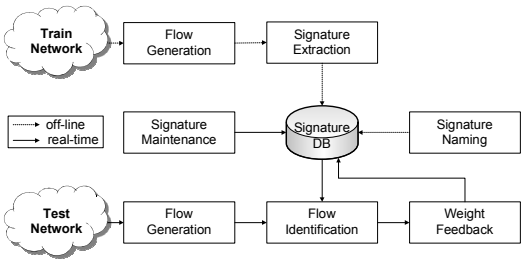

Fig. 1. System architecture of identification system

IV. SIGNATURE MANAGEMENT

The ultimate goal of signature maintenance is to maintain the signatures that analyze traffic over a long period of time. However, it is not easy to predict whether the signature will be used in the future because signature maintenance takes place in real time. Therefore, we identify the characteristics correlated with the LD (Life Duration) of the signature, and define these as management elements. Further, we propose a maintenance function that use these defined management elements.

TABLE 1. TRAFFIC TRACE

Date	2012.03.01~2012.03.31
Local IP	10,494
Flow(K)	659,694
Byte(G)	47,848

In order to define a management element, we collected traffic data from a campus network for a month (KU-Cam-01: shown in table 1). We then applied the identification system without using the maintenance method. To understand the characteristics of the generated signature, we renewed the signature feature value based on statistical information of identified traffic.

Figure 2 shows a histogram of the LD of the generated signature. LD denotes the time period from the beginning to the end, during which the signature analyzes traffic. The number of signatures that have a large LD (longer than two weeks) accounts for 6.41% (a) of the total number of signature. On the other hand, 53.41% (b) of the signatures showed an LD of less than one minute, which is the minimum unit of measurement for time. This indicates that most of the signatures are used in traffic identification only when they are created; they are no longer used after that. We define a signature that has an LD of less than one minute as a "flash signature". In this chapter, we analyze the characteristics of flash signatures.

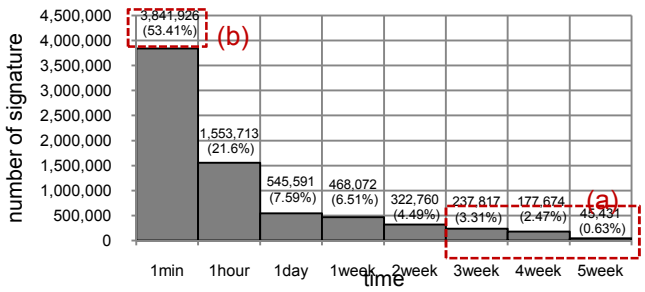

Fig. 2. Histogram of signature LD (Life Duration)

Table 2 shows the amount of the traffic identified by flash and non-flash signatures respectively. Flash signatures, which account for 53.41% of all signatures, identify the traffic in 1.26% of total flows and 0.96% of total bytes. To sum up, flash signatures form the majority of created signatures but identify considerably less traffic.

TABLE 2. COMPARISON OF FLASH SIGNATURES AND NON-FLASH SIGNATURES

Name	Flash Signature	Non-Flash Signature	Total
Num. of Signatures	3,841,926 (53.41%)	3,351,058 (46.58%)	7,192,984 (100%)
Identified Flows	3,354,672 (1.26%)	261,761,224 (98.73%)	265,115,896 (100%)
Identified Bytes(M)	282,336 (0.96%)	29,093,558 (99.03%)	29,375,895 (100%)

In this paper, we define maintenance elements that can be used in a maintenance function. Based on the above experiments, we propose CPC (Counter Peer Count), FC (Flow Count), and BC (Byte Count) as the maintenance elements to be used. hs(CFC) indicates the number of clients in identified traffic. hs(FC) and hs(BC) represent the number of flows and bytes, respectively, which are identified traffic by using the signatures. We suggest a maintenance function for calculating the weight of signatures for maintaining the signature. This function uses the maintenance elements as follows:

$$M_{proposed}(hs) = M_t(hs) + M_{t-1}(hs) - 1 \quad (3)$$

$$M_t(hs) = \left(\alpha \cdot \frac{min(hs_t(CFC), th_{CFC})}{\min(HS_{max,t}(CFC), th_{CFC})}\right) + \left(\beta \cdot \frac{min(hs_t(FC), th_{FC})}{\min(HS_{max,t}(FC), th_{FC})}\right) + \left(\gamma \cdot \frac{min(hs_t(BC), th_{BC})}{\min(HS_{max,t}(BC), th_{BC})}\right) \quad (4)$$

The proposed maintenance function re-calculates the weight value by adding the current estimated value at each management interval. The feature values of signatures accumulated from the beginning are not used. In addition, the function decreases the weight by 1 each interval to avoid the influence of the old value on the current weight. If the value of the weight is under 0, the signature is deleted from the DB. Also, several threshold values (α, β, and γ) are used to help measure reliable weight.

V. EXPERIMENT AND RESULT

To evaluate the performance of the proposed maintenance function, we define evaluation metrics and operate our identification system for the duration of one month of traffic data (KU-Cam-01). We focus only on the performance of the maintenance function, omitting the application naming part. Additionally, for objective evaluation of the experiment, we compare the proposed method with the accumulated and timeout methods. The accumulated method is not applying maintenance method, and the timeout method [3] involves deleting unused signature when the weight becomes under 0, as given in equation (5).

$$M_{timeout}(hs) = \alpha - \left(current\ time - hs(LT)\right) \quad (5)$$

TABLE 3. EXPERIMENTAL RESULT

Maintenance Method	Completeness		Num. of Signatures	Average LD (Life Duration)
	Flow (%)	Byte (%)	Avg. (K)	Avg. (sec)
Accumulated method	79.45	95.64	18,159	24,827
Proposed method α, β, γ =60	62.34	90.92	119	150,904
Timeout method α =60	59.58	89.41	60	4,759

Table 3 shows the experimental results. In terms of completeness, the accumulated method has good performance because all the signatures created during the test are accumulated without deletion. However, when compare with the proposed method, the number of signatures is about 150 times larger. The proposed method, when compare with the timeout method, shows better performance in term of completeness and average LD. The difference of average LD is about 30 times. This means that the proposed method can maintain a higher number of useful signatures in the DB.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed an efficient method for maintenance of header signatures that could prove useful in traffic identification. For this, we defined maintenance elements after analyzing the features of signatures, and proposed a maintenance function based on these elements. We proved the performance of the proposed method by carrying out empirical tests. Via experiments, we found that the proposed method has better performance than the timeout method, even the accumulated.

In future research, we plan to define more maintenance elements and propose several maintenance functions using these elements.

REFERENCES

[1] Sung-Ho Yoon, Jun-Sang Park, Myung-Sup Kim, "Signature maintenance for Internet application traffic identification using header signatures," Proc. of the 4th IEEE/IFIP International Workshop of the Management of the Future Internet (ManFI 2012), Hwaii, USA, Apr. 16, 2012.

[2] A. Moore, K. Papagiannaki, "Toward the Accurate Identification of Network Applications," Proc. PAM 2005, Boston, USA, 2005.

[3] M. Baldi, A. Baldini, N. Cascarano, and F. Risso, "Service-based traffic classification: Principles and validation", Proc. of the IEEE 2009 Sarnoff Symposium, Princeton, NJ, USA, Mar. 2009