Method for Resolving Traffic Classification Limitations caused by Traffic Collection Points

Hyun-Min An, Jae-Hyun Ham, and Myung-Sup Kim

***Abstract*—**Application-level traffic classification is essential for effective network management and stable service provision. Recently, traffic classification methods that are based on the statistical information in a flow have been proposed for application-level traffic classification. The packet transmission order, packet transmission direction, and the packet size in a flow are used as statistical information. However, differences in traffic collection points cause inconsistencies in the statistical information of the flow. Furthermore, the analysis results cannot be trusted. Therefore, in this paper, we analyze the limitations on traffic classification caused by traffic collection points, and we propose a novel method for resolving these limitations. The proposed method is verified through experiments conducted in a campus network.

***Keywords*—**Full-duplex TCP session, statistical flow information, traffic classification, traffic collection

I. Introduction

ENTERPRISE and campus networks typically impose a set of policies, such as quality of service (QoS) and service-level agreements (SLAs), for the efficient management and operation of network resources. For example, schools and public institutions implement policies to control traffic that consumes excessive network resources and that is not related to the organization's purposes, such as peer-to-peer (P2P) and game traffic. For these policies, fast and accurate traffic classification in the application layer is essential. Recently, several methods have been introduced that use statistical flow information [1][2]. However, traffic classification methods that use statistical flow information as their main feature have limitations caused by traffic collection points.

In this paper, in order to resolve said limitations, we propose a method for reordering packets by comparing the packet sequence and the packet acknowledgement number. Through experiments performed in a campus network, we detected a number of abnormal sessions, which indicated that the limitations described in the previous paragraphs must be resolved in order to accurately classify traffic.

The remainder of this paper is organized as follows. Section II briefly reviews and summarizes previous work. Section III introduces the reasons for the limitations caused by traffic collection points. Section III also introduces the methods for detecting and resolving these limitations. Section IV describes our experimental method and analyzes detection results. Conclusions and future work are presented in Section V.

Hyun-Min An is with the Dept. of Computer and Information Science, Korea Univ., 2511, Sejong-ro, Jochiwon-eup, Sejong, South Korea (e-mail: queen26@korea.ac.kr).

Jae-Hyun Ham is with the Dept. of Computer and Information Science, Korea Univ., 2511, Sejong-ro, Jochiwon-eup, Sejong, South Korea (e-mail: jhham@korea.ac.kr).

Myung-Sup Kim is with the Dept. of Computer and Information Science, Korea Univ., 2511, Sejong-ro, Jochiwon-eup, Sejong, South Korea (corresponding author to provide phone: +82-44-860-1378; fax: +82-44-860-1584; e-mail: tmskim@korea.ac.kr).

II. Related Work

In today's Internet environment, there are various applications; therefore, classifying traffic at the application level is not easy. In order to accurately classify such traffic, several methods have been proposed. Traditional methods can be divided into three types: signature-based classification [2], classification based on the correlation of traffic [4], and machine learning-based classification [5,6].

The targets of this paper are the signature-based and machine learning-based classification methods. In this section, we explain traffic classification methods using statistical flow information [2], which our study briefly advanced.

We used statistical information that consists of transmission order, transmission direction, and payload size of packets in a flow. The direction is expressed as two values, positive (+) or negative (–). In the case of the transmission control protocol (TCP), the + sign means that the packet transfers from the client to the server, and the – sign means that the packet transfers from the server to the client. Because distinction between the server and the client is not clear in the user datagram protocol (UDP), the meaning of +/– merely indicates that the directions are opposite. Therefore, we decide that the first packet's direction is +, and the following packets obtain their direction value through comparison with the first packet's direction. Packets with the same direction are +; otherwise, the packets are –. The payload size of a packet represents the payload length in a packet that has payload data.

We use vector representation to express the feature explained in the previous paragraph. We assume that f_k is a k-th input flow. The $d_{k,i}$ and $s_{k,i}$ are the transmission direction and payload size of the i-th packet of the k-th flow, respectively. The $d_{k,i}$ can have two direction values: + and –. v_k is a function that expresses f_k into an N-dimensional vector. It can be written as expressed in (1).

$$v_k = \{d_{k,1} \times s_{k,1},\ d_{k,2} \times s_{k,2},\ \ldots,\ d_{k,N} \times s_{k,N}\} \qquad (1)$$

For flow grouping and traffic classification, we need a measurement of the distance between two vectors as the distance similarity measurement. In our method, we use the distance per dimension; thus, the distance between two flow vectors is represented by distance vector d . The i -th element of the distance vector, d_i , is the difference between the i -th element of two flow vectors. The distance vector d , which is the distance between v_k and v_j , can be expressed as indicated by (2) where $v_{k,i}$ represents the i -th element of the k -th flow vector.

$$d(v_k, v_j) = \{|v_{k,1} - v_{j,1}|, |v_{k,2} - v_{j,2}|, \ldots, |v_{k,N} - v_{j,N}|\} \quad (2)$$

Our method has been divided into two main parts: signature generation and traffic identification. First, the signature generation part converts flows into feature vectors that consist of statistical flow information. The vectors are divided by process. The process flows are entered into the flow-grouping step and grouped by the distance similarity of flow vectors. Then, our method optimizes the groups of all processes and generates the signature in each group.

The traffic identification part generates flow through the use of packets that are collected continuously. Next, the method converts the flow into vectors and matches a flow vector with its corresponding signature in order to identify the correct application name.

III. Limitations Caused by Traffic Collection Points

In traffic classification methods that use statistical flow information, feature consistency is extremely important, because these methods extract unique patterns from the traffic statistical information, and classify the traffic by comparing the statistical information of the traffic with the unique patterns. Therefore, when unique patterns are extracted without resolving this limitations, and are applied to points different from the extraction point, the classification results cannot be trusted.

In this section, we explain the lack of feature consistency caused by traffic collection points, and propose methods for detecting and resolving such inconsistencies.

A. The lack of the feature consistency

In a full-duplex TCP session, one end host does not wait for another host to finish transferring data, and begins transferring its own data as the other host is still transferring. Accordingly, packets that are transferred by the two end hosts cross in the middle of the transmission path. At such a moment, feature inconsistency occurs.

For example, if a client and a server communicate as shown in Fig. 1, the traffic collection points, from C1 to C4, have a different packet transmission order. Therefore, the statistical information from each traffic trace collected by each collection point is different from the others. Consequently, a lack of the feature consistency has occurred.

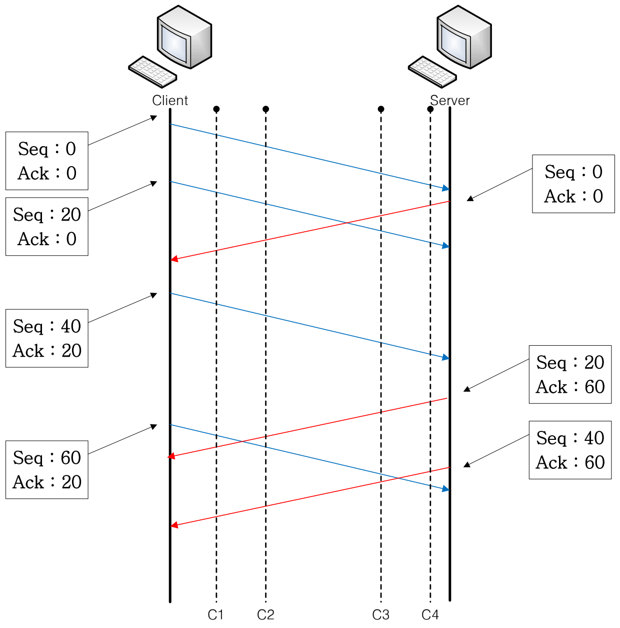


Fig. 1 Lack of Feature Consistency

B. Detecting Algorithm

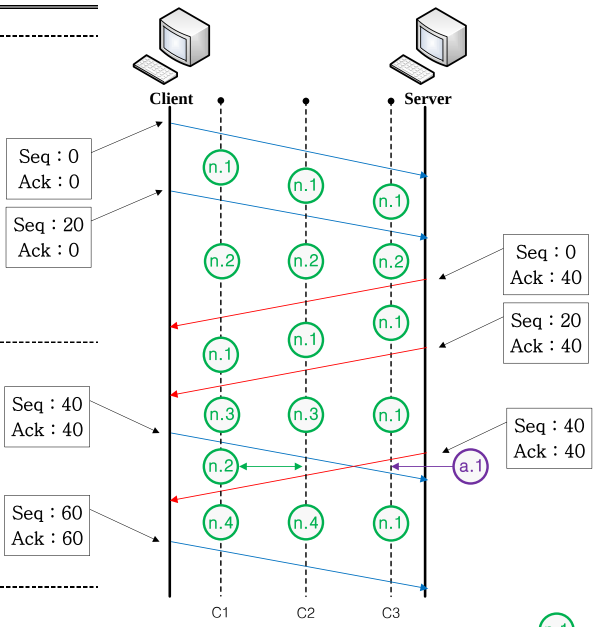
For full-duplex TCP sessions, there is no correct order for packet transmission; there is only the accepted order in which end hosts send and receive packets. Therefore, a standard packet order must be established. In this paper, we choose the packet order accepted by the client host as the standard, because most TCP sessions start with a request from the client to the server, and continue with a response from the server to the client. In other words, the session changes continuously according to requests from the client. Thus, in this paper, and regardless of the traffic collection point location, we reorder the packets in a flow according to the standard order.

Fig. 2 shows the pseudo code for the detecting algorithm. The algorithm's input data is the TCP session that has payload packets. The payload packet is the packet that has payload data. P(n) indicates the n-th packet in the session. P(n)(Seq) and P(n)(Ack) are the sequence number and acknowledgement number of the n-th packet in the session, respectively. The forward packet is the packet transferred from the client to the server, and the backward packet is the packet transferred from the server to the client. Our proposed method detects normal TCP sessions and abnormal TCP sessions through four detecting conditions that are derived by comparing P(n)(Seq), P(n)(Ack), P(n+1)(Seq), and P(n+1)(Ack).

Before we explain all the cases using the subsequent figures in this paper, we must define the symbols used in the figures. For the normal cases, we use symbols from "n.1" to "n.4"; for the abnormal cases, we use symbols from "a.1" to "a.4." The symbols from "C1" to "C3" represent the three traffic collection points. The packet cross indicates that the forward packet and the backward packet crossed before arriving to their destination.

```
Remove all non-payload packets from the packet sequence
1: procedure Detect Normal Packet Sequence
2:   Input : packet sequence of a TCP flow
3.   if ( P(n) (direction) == P(n+1) (direction) ) then normal
  4:   if ( P(n) == Fp && P(n+1) == Bp) then normal
  5:   if ( P(n) == Bp && P(n+1) ==Fp) {
  6:     if ( P(n)(Ack) <= P(n+1)(Seq) && P(n)(Seq) < P(n+1)(Ack) )
  7:     then normal
  8:     else abnormal
  9:   }
10: end procedure
1: procedure Detect abnormal Packet Sequence
2:   Input : packet sequence of a TCP flow
3:   if ( P(n) ==Bp && P(n+1) ==Fp ) {
4:     if ( P(n)(Ack) <= P(n+1)(Seq) && P(n)(Seq) >= P(n+1)(Ack) ) }
5:     then abnormal
6:     else normal
7:   }
8: end procedure
Payload packet : a packet with payload data
Non-Payload packet : a packet without payload data
P(n): n-th payload packet in a TCP flow
Forward packet (Fp) : a packet from client to server
Backward packet (Bp) : a packet from server to client
```

Fig. 2 Pseudo code for the Detecting Algorithm



- **if (P(n) (direction) == P(n+1) (direction)** n.1
- **if (P(n) = Fp && P(n+1) = Bp)** n.2
- **if (P(n)(Ack) == P(n+1)(Seq) && P(n)(Seq) < P(n+1)(Ack))** n.3
- **if (P(n)(Ack) < P(n+1)(Seq) && P(n)(Seq) < P(n+1)(Ack))** n.4

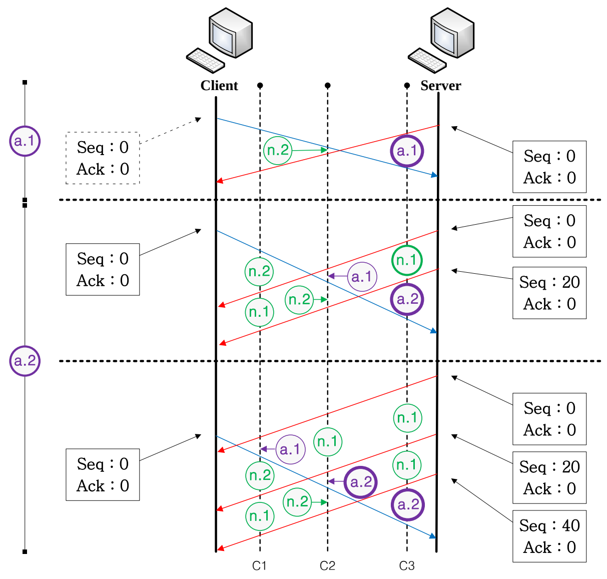
Fig. 3 Normal Case

Fig. 3 shows the examples and the detecting condition for four normal cases. Normal Case 1, the n.1, occurs when P(n) and P(n+1) have the same transmission direction. Normal Case 2, the n.2, occurs when P(n) is the forward packet and P(n+1) is the backward packet. When P(n) is the forward packet, we always have a normal case, because the packet order for each traffic collection point is the same as the standard packet order, which is the client's packet order. In Normal Cases 3 and 4, P(n) is the backward packet, so we must compare P(n) to P(n+1) to detect normalcy.

Normal Case 3, the n.3, indicates that P(n) is the backward packet, but there are no packets crossing. Therefore, in the n.3, regardless of the traffic collection point, the collected packet order is fixed. In order to detect the n.3, a comparison between the value of the sequence and the acknowledgment number of P(n) and P(n+1) is essential.

Normal Cases 1, 2, and 3 are detected when no packets are crossing. However, Normal Case 4 is detected after the packets cross. In other words, if Normal Case 4 is detected on a traffic collection point, there is at least one Abnormal Case 1 on the other collection point. Normal Case 4, the n.4, has a condition in which P(n) is the backward packet and P(n+1) is the forward packet. In addition, there is a packet cross before the n.4. Consequently, in order to detect the n.4, a comparison between the value of the sequence and the acknowledgment number of P(n) and P(n+1) is essential, just as it is for detecting the n.3.

In the example shown in Fig. 3, C1 and C2 collect packets in the order accepted by the client. Conversely, the traffic collection point C3 collects packets in a different order from the client. Therefore, the unique patterns extracted from C1 or the C2 cannot be used on C3 with fine accuracy.



- **if (P(n)(Ack) == P(n+1)(Seq) && P(n)(Seq) == P(n+1)(Ack))** a.1
- **if (P(n)(Ack) == P(n+1)(Seq) && P(n)(Seq) > P(n+1)(Ack))** a.2

Fig. 4 Abnormal Case 1, 2

Similarly, the unique patterns extracted from C3 cannot be accurately used on C1 or C2. Consequently, in order to collect the traffic, this problem must be resolved.

Fig. 4 shows Abnormal Cases 1 and 2, which are the a.1 and the a.2. In Fig. 4, the packet order of each traffic collection point, from C1 to C3, is different from each other. In the a.1 case, P(n) is the backward packet and P(n+1) is the forward packet. In addition, there is essentially one packet cross. When the a.1 occurs, this abnormal case is quite simple to resolve, because one packet cross creates two packet orders.

However, when the a.2, the a.3, or the a.4 occur, there are two or more packet crosses, which create many packet orders; consequently, the resolution of these problems is considerably complicated. Abnormal Case 2, the a.2, occurs when two packet crosses are caused by one forward packet and two backward packets. Before the a.2 occurs, the a.1 always happens first.

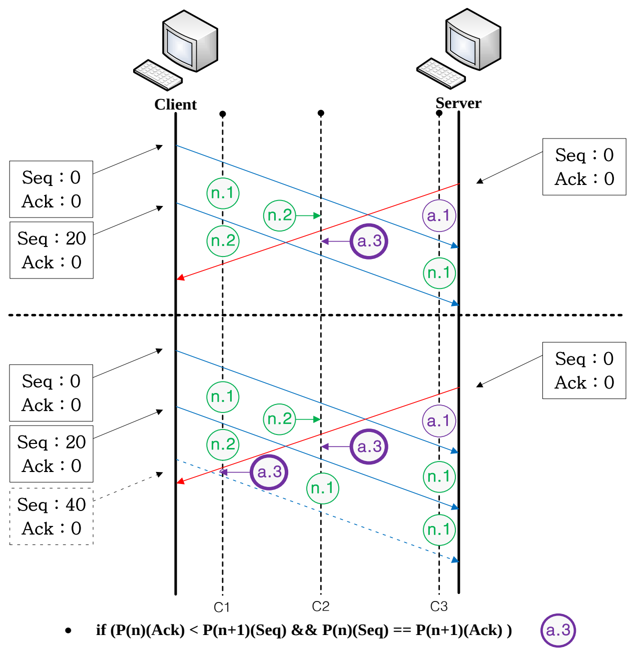


Fig. 5 Abnormal Case 3

Fig. 5 shows Abnormal Case 3, the a.3. The a.3 occurs when two packet crosses are caused by one backward packet and two forward packets. Before the a.3 occurs, the a.1 always happens first, as is the case with the a.2.

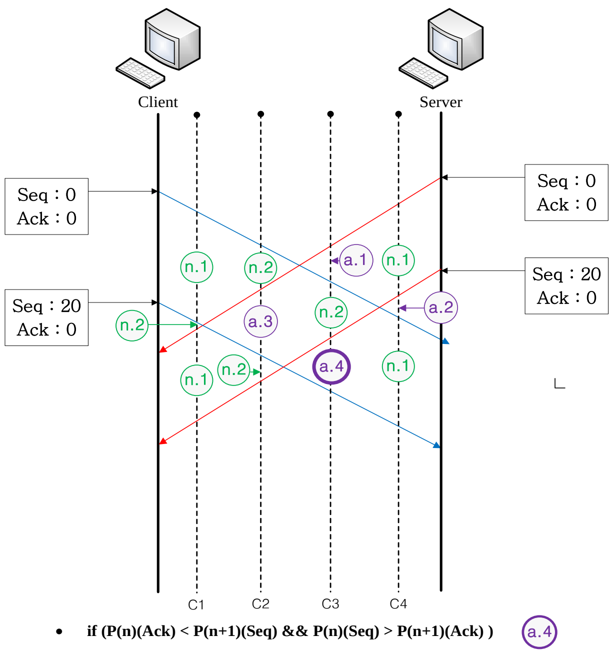


Fig. 6 Abnormal Case 4

Abnormal Case 4, the a.4, is represented in Fig. 6. The a.4 is detected when two or more forward packets and backward packets are crossed. The a.1, the a.2, and the a.3 occur before the a.4. Many abnormal cases can occur at the same time; therefore, a method that considers all cases is required in order to resolve these problems.

C. Solving Algorithm

Fig. 7 shows the pseudocode for the solving algorithm. When abnormal cases are detected in the traffic traces, the algorithm exchanges the P(n) order for the P(n+1) order. When a packet cross has occurred once, switching the packet order is required only once to resolve the problem. However, if there are continuous packet crosses, switching is required from the first packet cross to the last. To handle this problem, when a packet cross is detected, the algorithm exchanges the P(n) order for the P(n+1) order, and backtracks to the top of the packet cross problem until no abnormal cases are detected.

```
Remove all non-payload packets from the packet sequence
1: **procedure** Resolve Abnormal Packet Sequence
2:   Input : packet sequence of a TCP flow
3.   **if** ( P(n) and P(n+1) are abnormal sequence ) {
4:      change P(n) and P(n +1);
5:      n = n - 1;
6:      **if** ( P(n) and P(n+1) are abnormal sequence ) **goto** 4:
7:   }
8: **end procedure**
```

Fig. 7 Pseudocode for the Solving Algorithm

Fig. 8 shows the flowchart for the proposed detecting and solving method. Two algorithms are used as a stream.

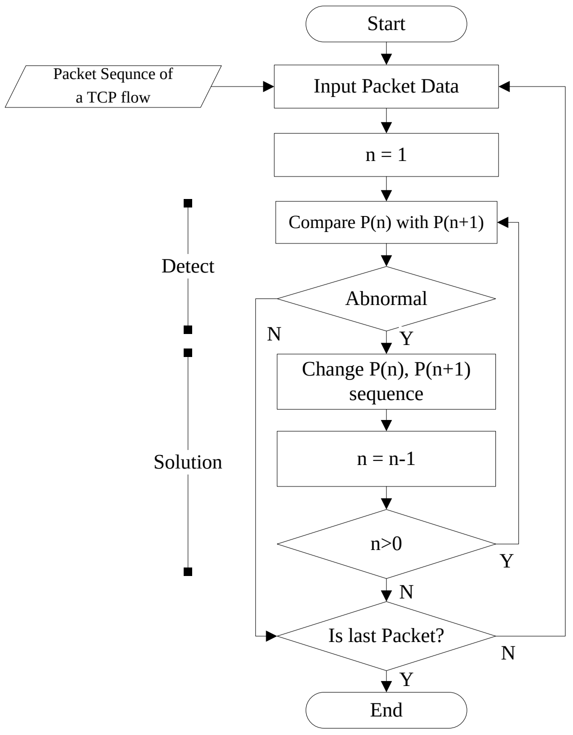


Fig. 8 Flowchart of the Detecting and Solving Method

IV. Results and Analysis

In this section, we describe the results of the experiment that analyzes the packet cross problem. This experiment shows the frequency with which the packet cross problem occurs in real traffic traces. For our evaluation, we collected bi-directional packet traces from the Korea University campus network. The campus network is configured with one router at the Internet junction; thus, we collected the traffic traces from the router using port mirroring. The one-day traffic, 2012-12-17 traffic, was collected by KU-MON [7]; the traces consist of the TCP sessions only.

Table I and II list the total detection amount of packet crosses. The total flow and packet are the average number of flows and packets that occurred in a minute. The flow detection indicates the amount of flows where there is one or more packet cross in the flow. The packet detection indicates the number of packet crosses. When we analyzed the amount of bytes, we measured it by adding the two packets that were involved in the packet cross.

TABLE I
ABNORMAL RATE CASE BY CASE

Case	Measure	Total	Detection	Bytes	Rate
a.1	Flow	16,282	349	3,345,013,424	2.141%
	Packet	50,584	95		0.187%
a.2	Flow	16,282	24	1,625,395,644	0.148%
	Packet	50,584	7		0.014%
a.3	Flow	16,282	7	332,020,732	0.042%
	Packet	50,584	2		0.004%
a.4	Flow	16,282	2,932	932,006,201,946	18.007%
	Packet	50,584	7,899		15.616%

TABLE II
TOTAL ABNORMAL CASE

Measure	Total	Detection	Bytes	Rate
Flows	16,282	3,120	937,308,631,746	19.162%
Packets	50,584	8,003		15.821%

The flow detection rate is 19.2%, and the packet detection rate is 15.8%. From the aspects of the flow, we can see that the abnormal flow rate is one-fifth of the normal flow rate. Consequently, solving the abnormal flow rate is essential for a trustworthy traffic classification.

V. Conclusion

In this paper, we analyzed the limitations caused by traffic collection points. Then, we proposed a method that detects and solves the limitations caused by traffic collection points, namely, the packet cross problem. The results of the experiment performed for evaluation purposes indicate that one-fifth of the traffic flow demonstrate this problem. Until such problem is solved, the traffic classification results of methods based on statistical flow information cannot be trusted, because the patterns used in these methods to collect traffic from a certain point cannot be applied to other traffic collection points.

Our future studies will focus on the implementation and verification of the solving algorithm. We will also research other problems that can affect statistical flow information, as well as analyze abnormal TCP sessions for the robustness of the traffic classification method.

Acknowledgment

This research was supported by BK21 Plus Program and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2007483).

References

[1] Young-Tae Han and Hong-Shik Park, "Game Traffic Classification Using Statistical Characteristics at the Transport Layer," ETRI Journal, Vol.32, No.1, Feb., 2010, pp.22-32.

[2] Hyun-Min An, Jae-Hyun Ham, and Myung-Sup Kim, "Application Traffic Classification using Statistic Signature," Proc. of the Asia-Pacific Network Operations and Management Symposium (APNOMS) 2013, Hiroshima, Japan, Sep. 25-27, 2013

[3] Liu, Hui Feng, Wenfeng Huang, Yongfeng Li, Xing "Accurate Traffic Classification," Networking, Architecture, and Storage, 2007. International Conference

[4] Myung-Sup Kim, Young J. Won, and James Won-Ki Hong, "Application-Level Traffic Monitoring and an Analysis on IP Networks," ETRI Journal, Vol.27, No.1, Feb., 2005, pp.22-42.

[5] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti, "Traffic Classification Using Clustering Algorithms," Proc. of SIGCOMM Workshop on Mining network data, Pisa, Italy, Sep. 2006, pp.281-286.

[6] Andrew W. Moore and Denis Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques," Proc. of the ACM SIGMETRICS, Banff, Canada, Jun., 2005.

[7] Jun S. Park, Jin W. Park, Sung H. Yoon, Young S. Oh, Myung S. Kim, "Development of signature Generation system and Verification Network for Application Level Traffic Classification," in Proc. KIPS conf., PuSan, Korea, Apr. 23-24, 2009, pp. 1288-1291.