An Integrated Method for Application-level Internet Traffic Classification

Mi-Jung Choi[1], Jun-Sang Park[2], and Myung-Sup Kim[2]
[1] Dept. of Computer Science, Kangwon National Univ.
1 Kangwondaehak-gil, Chuncheon-si, Gangwon-do, 200-701, Korea
[e-mail: mjchoi@kangwon.ac.kr]
[2] Dept. of Computer and Information Science, Korea Univ., Sejong Campus
2511, Sejong-ro, Sejong City, 339-700, Korea
[e-mail: jungsang_park@korea.ac.kr, tmskimg@korea.ac.kr]
*Corresponding author: Myung-Sup Kim

Received July 18, 2013; revised October 21, 2013; revised January 11, 2014; accepted February 9, 2014; published March 31, 2014

Abstract

Enhanced network speed and the appearance of various applications have recently resulted in the rapid increase of Internet users and the explosive growth of network traffic. Under this circumstance, Internet users are eager to receive reliable and Quality of Service (QoS)-guaranteed services. To provide reliable network services, network managers need to perform control measures involving dropping or blocking each traffic type. To manage a traffic type, it is necessary to rapidly measure and correctly analyze Internet traffic as well as classify network traffic according to applications. Such traffic classification result provides basic information for ensuring service-specific QoS. Several traffic classification methodologies have been introduced; however, there has been no favorable method in achieving optimal performance in terms of accuracy, completeness, and applicability in a real network environment. In this paper, we propose a method to classify Internet traffic as the first step to provide stable network services. We integrate the existing methodologies to compensate their weaknesses and to improve the overall accuracy and completeness of the classification. We prioritize the existing methodologies, which complement each other, in our integrated classification system.

Keywords: application-level traffic classification, signature-based classifier, behavior algorithm, correlation algorithm, integrated classification

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).

http://dx.doi.org/10.3837/tiis.2014.03.007

1. Introduction

Recently, voice, video, and mobile services are being provided over converged IP networks, and these services generate much network traffics [1]. The introduction of various services and applications to the market has caused enterprises and individual users to become highly dependent on the Internet. Internet users are eager to receive reliable and QoS-guaranteed services. To provide reliable network services, network managers need to perform control measures involving dropping, blocking or prioritizing each traffic type. It is also necessary to rapidly measure and correctly analyze Internet traffic and to classify network traffic according to applications. The traffic classification result also provides basic information for ensuring service-specific QoS. Moreover, traffic monitoring and measurement for efficient network operations and management are fundamental to understanding current network usage patterns and to planning network expansion.

The importance of network traffic monitoring and analysis is growing in various areas, such as for guaranteeing QoS and security. The first and most important step of network traffic monitoring and analysis is to determine the application-level identities of Internet traffic and to classify them into specific categories for analysis purposes. Accurate classification of application-level traffic is an important factor in determining reliability of analysis of interrelated utilization such as usage-based charge by application, application-based traffic control, support for Service Level Agreement (SLA), application-level traffic security. To date, there have been many studies [2][3][4][5] on the methods for classifying application-level traffic. However, although they have strengths in some aspects [2][6], most of these methods are difficult to use as a single method in a real-time traffic classification system for large-volume, diverse-type, high-speed Internet traffic. Additionally, there is no single method for achieving optimal performance in terms of accuracy, completeness, and applicability in an actual network environment.

In this paper, we propose an integrated method for classifying Internet traffic in real-time. We combine several different methods in cascading and parallel steps to achieve satisfactory performance in an operational network environment. To enhance classification completeness and accuracy, we divide the classification process into four cascading steps: flow generation, individual classification, flow integration, and flow correlation. The flow generation step combines and merges raw packets into flows and delivers them to the next step. In the individual classification step, we simultaneously operate several different classification methods which were developed in our previous research [2][8][9]. In the flow integration step, we merge the multiple identification results from the previous steps into one result. In the flow correlation step, we finally identify unknown traffic flows by utilizing relational information among traffic flows. To prove the validity and feasibility of the proposed classification system, we designed and implemented the proposed system, and deployed it in our campus network.

The rest of this paper is organized as follows. In Section 2, we briefly summarize related work discussing previous traffic classification methods. In Section 3, we outline some important considerations required for a real-time traffic classification system. In Section 4, we describe the proposed classification algorithm and the system implementation. In Section 5, we present the Internet traffic classification results by applying the system to an enterprise network and analyzing the results. In Section 6, our conclusions and the direction of related future work are discussed.

2. Related Work

Internet traffic classification methods can be largely categorized into port-based [6][10][11], signature-based [7][12], machine-learning-based [13][14], and traffic-correlation-based [2][4] methods. Each of them has their own merits as well as certain limitations.

Internet traffic classification has used port information defined in IANA [10], such as HTTP, FTP, and SMTP, which use well-known port numbers. However, other recent applications commonly use well-known port numbers to pass through firewalls and intrusion prevention system (IPS). Accordingly, port-based classification no longer provides highly reliable and complete analysis results.

Signature-based classification is a method of identifying unique patterns used by only one application; therefore, the given application can be differentiated from other ones, and its unique signature can be used to classify it. This method offers the advantage of accurately classifying applications by their identified signatures. However, it has limitations in that the signatures of all applications must be extracted in a tedious manual process. There are applications whose signatures are difficult to extract and newly appearing applications are not handled well.

Machine learning-based classification is a method of performing classification after a machine learning algorithm has learned the factors, such as port, inter-arrival time, packet size, etc., that characterize each application. An advantage of this method is that the classification accuracy is high compared to other methods because traffic is classified using an advanced algorithm. The disadvantage, however, is that application traffic with a limited scope should be collected and learned during preparation, which involves a certain amount of overhead. When this method is applied to an actual network, the classification accuracy is low for traffic that has not yet been learned. At present, there are several studies that attempt to classify real-time traffic using machine-learning-based methods [15][16].

Traffic-correlation-based classification is a method of classifying traffic by finding correlation information among applications. For example, the correlations are presented in weighted values based on unique characteristics, such as the three-level address system (IP address, port number, and protocol) of Internet traffic and traffic generation. In conjunction with certain thresholds, these correlations are used to identify applications. In terms of classifying traffic, this method has the advantage of optimum completeness because it utilizes the characteristics of applications. However, when it is applied to actual Internet traffic, it is difficult to guarantee the reliability of the classification results because thresholds are found by trial and error without definite algorithms for utilizing the characteristics of each application.

The integrated method for application-level traffic classification proposed in this paper is an extension of the signature-based classification methodology. This classification methodology is constructed by appropriately combining header, statistic, and payload signature classification methods and behavior and correlation algorithms. Using correlations among traffic, the method can classify the traffic that each system cannot independently analyze.

3. Considerations

In this section, we discuss some important considerations for the design and implementation of the proposed real-time traffic classification system. First, the traffic classes by which the classification system determines Internet traffic identities must be defined. The degree of traffic classes can be defined with various ranges depending on the analysis purpose, such as

protocol level, application level, service level, and so on. In this paper, we conduct traffic classification at the application level, where control of specific application traffic is possible. For example, diverse applications, such as Internet Explorer (Web browser), uTorrent (P2P application), Nateon (messenger application), and Melon (multimedia application), transmit data using the HTTP protocol. In this case, if a specific P2P application must be blocked at the firewall, the blocking control can be achieved only when application-level traffic classification is possible.

Compared to existing studies [2][8], the flow format is changed; in this study, classification is conducted with two-way flows instead of one-way flows. One-way flow is a set of consecutive packets with the same 5-tuple information (source IP, destination IP, source port, destination port, and L4 protocol) in one direction. Two-way flow consists of a request flow and response flow, which are two one-way flows of opposite directions. Two-way flow is divided into bidirectional flow or unidirectional flow, depending on whether a pair of the relevant flow exists or not. A typical advantage of two-way flow is reduction in the number of flows. It is shown that the number of flows is reduced by almost half. Another advantage of this format change is the easy processing of reverse flow. In existing studies [5][6][12], as one-way flows were used, it was necessary to find the reverse flow in order to apply the relevant analysis results to the reverse flow after analyzing one side. However, these overheads can be reduced because this process is unnecessary in a two-way flow format.

Classification scope is also an important consideration. We aim to classify all traffic of a target link at the application level. In this case, application-level classification information for all traffic generated from a target link can be provided. However, if high accuracy is not guaranteed, the classification results cannot be trusted. Accordingly, to obtain reliable results in the application-level analysis, a method to verify the classification result should first be established. To prove whether the classification result is correct or not, we have developed the Traffic Measurement Agent (TMA) and Traffic Measurement Server (TMS) [8]. The TMAs are installed at the end hosts and periodically collect socket information used by the processes running at the end hosts. The TMAs then send the information to the TMS. Using TMA information, we can generate ground-truth traffic information to judge the performance of a traffic classification system. The TMS logs the information received from the TMAs, compares it with the data analyzed by a classification system, and checks whether the classifications were correct.

The last consideration is classification time. It is divided into on-line processing or off-line processing, depending on the traffic classification time. Off-line processing is widely used because it makes the system architecture simple and system load problems can be easily solved. However, there are limitations in utilizing the analysis results because the results cannot be immediately obtained [17][18][19][20]. An on-line processing system requires distributed processing because the workload is heavy, thus this causes problems in terms of complicated architecture. However, on-line analysis is absolutely required for providing differentiated service. In this study, traffic is classified by the on-line processing method.

4. Integrated Traffic Classification System

In this section, we propose an Integrated Traffic Classification System (ITCS) as a means to enhance the completeness and accuracy of application-level traffic classification and compensate the existing classification methodologies. **Fig. 1** shows the overall ITCS architecture. ITCS is comprised of four levels. At the first level, all packets collected from a target link are converted by a Flow Generator (FG) into a two-way flow. The generated flows

are used as input data for the four classification systems running simultaneously at the second level. The four classification systems are comprised of three signature-based classification systems (header, statistic, and payload signature classifier), and one behavior-based classification system. The signature-based classification systems perform application-level identification through signature matching, which compares each input flow with predefined signatures [8]. The behavior-based classification system also classifies flows using an algorithm based on the behavior analysis of specific applications. For example, the Skype and uTorrent applications can be identified by the behavior-based analysis system.

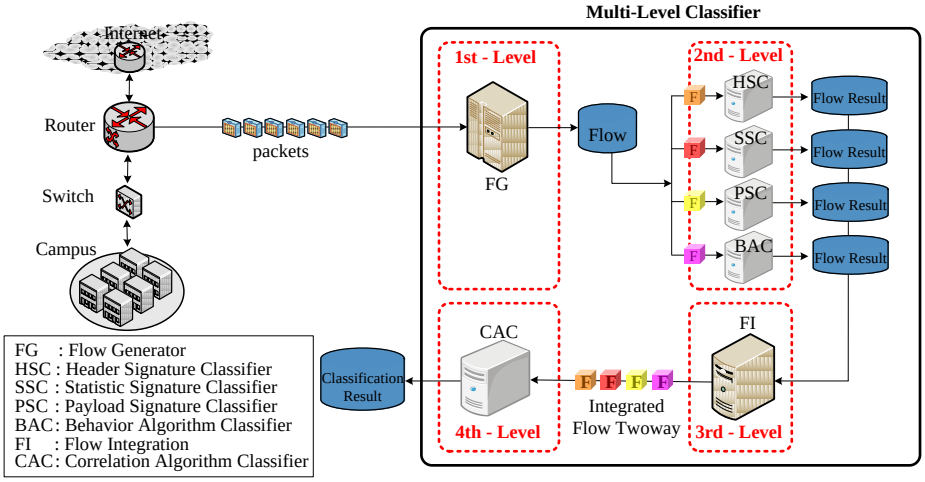

Fig. 1. Integrated Traffic Classification System Architecture

The flows that are classified as fitting certain applications, as well as the flows that are not classified, are integrated by Flow Integration (FI) at the third level. Every flow generated at the first level enters each analysis system at the second level. Therefore, because four classification systems simultaneously analyze a flow at the second level, an identical result would be produced in the best case, and four different classification results would be produced in the worst case. The FI stage at the third level is a process that selects the best classification result from the analysis result produced by four different classifiers. At this point, with the unclassified results aside, it selects a classification result based on the classifier's priority. The classifier's priority is behavior algorithm classifier (BAC)> payload signature classifier (PSC)> statistical signature classifier (SSC)> header signature classifier (HSC), and its selection criterion is defined on the basis of each classifier's accuracy. At the fourth level, a correlation algorithm classifier (CAC) additionally classifies the flows, which have not been classified at the second level, by using the three correlation algorithms with inputs from the flows integrated at the previous (third) level. The classification algorithms are described in the following subsections.

Many identification methods [21] have been performed using the first several packets of an entire flow for flow control. The proposed ITCS is designed to be able to work in on-line processing mode for the flow control. As soon as a two-way flow is created with the first few captured packets in a flow at the first level, the flow information is delivered to the second level before the completion of the flow. The four individual classification modules at the second level can identify the flow with the first several packets of a flow due to the signature properties of our ITCS. The third and fourth level modules are immediately activated after the classification results from the previous level are generated. Therefore, our ITCS can generate

classification result before all the packets in a flow are captured, which can be deployed in a real network environment.

4.1. Signature Extraction System

Fig. 2 shows the overall architecture of the signature extraction system, which marks a preliminary step for the classification. The Packet Collector (PC) collects all packets on the link and sends them to the FG, while the FG generates flows out of them. The TMS collects TMA log data from a number of end hosts in which the TMAs are installed, delivering the data to the Ground Truth Generator (GTG). The GTG generates ground truth traffic data for traffic classification by comparing the flows generated earlier by the FG. Based on this Ground Truth (GT) information, the Header Signature Generator (HSG), Statistic Signature Generator (SSG), and Payload Signature Generator (PSG) produce signatures for each application. These signature-extraction systems generate header, statistic, and payload signatures and store them in an XML file. The generated signatures are later used to perform application-level traffic classification as input data for the HSC, SSC, and PSC. The Behavior Algorithm Generator (BAG) also establishes a classification algorithm based on the GT. This will be described in Section 4.2.2. The classification results are used as input data for the correlation-based classification systems of the fourth level to extract the final classification result.

We first look into the header signature; it uses information in the IP and transport headers. From among the IP and transport header information, the header signature is comprised of a 3-tuple (IP address, port number, L4 protocol) used only in one application. For example, Internet applications pass through the process of user authentication before providing actual service, and this process is covered by a specific login server. Accordingly, the 3-tuple of the login server becomes one of the header signatures that can classify the correspondent application.

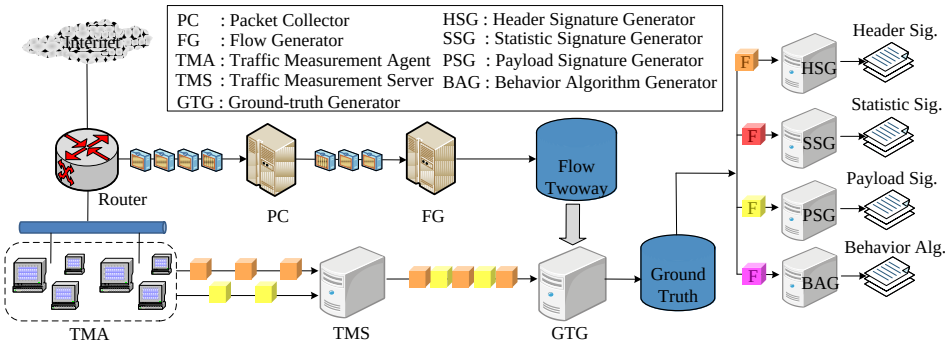

Fig. 2. Signature Extraction System Architecture

Fig. 3 is a fixed IP-port state transition diagram [9], which is the core of the automatic header signature extraction algorithm. Initially, 3-tuples of all servers fall under the "Discard" state. If a 3-tuple of the server from the already known application comes into the signature extraction system as input data, the relevant 3-tuple moves to a "New" state.

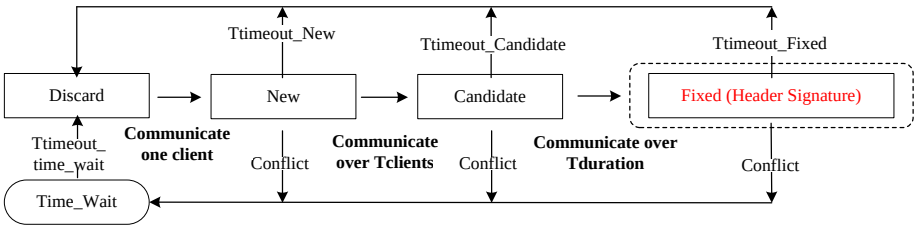

Fig. 3. Fixed IP-port State Transition Diagram for Header Signature Generation

If the number of requests sent by clients to the server of the relevant application exceeds the threshold "Tclienthost," the relevant application moves from the "New" state to the "Candidate" state. Lastly, if the time when client requests are received is not concentrated on a single moment, and another threshold, "Tduration," is instead exceeded, it becomes evident that requests for the server are distributed and that the relevant server is using a fixed IP and port. At this point, the application finally moves from the "Candidate" state to the "Fixed" state. The 3-tuples of the application servers that fall under the "Fixed" state become fixed IP ports and are elected as header signatures. To prevent 3-tuples of applications from continually remaining in a "New," "Candidate," or "Fixed" state, "Ttimelimit," which is the threshold of each state, is used to verify if there are any changes. If no request occurs during the threshold, they are returned to a "Discard" state. Each occurrence of an application's 3-tuple is observed to check whether that 3-tuple is shared by other applications. If a 3-tuple is used by different applications, it is defined as a "Conflict," and it moves to the "Time_Wait" state [9].

The statistic signature is the unique statistical feature of an application that can differentiate it from other applications; it is generated from its packet data (packet size, window size, etc.) or the capture information (packet capture time, packet inter-arrival time, etc.). The statistical information used in this paper is the packet size distribution; specifically, the payload size and packet direction. We disregard the non-payload packets in the statistic signatures. For example, the TCP control packets, such as SYN, RST, and FIN, do not include payload; therefore, these packets are not counted in the statistic signature construction. The direction of a packet is expressed in a positive or negative number. In the case of TCP, a positive number means a packet is moving from a client to a server, while a negative number means a packet is moving from a server to a client. As the server/client division is not clear in the case of UDP, the positive/negative numbers only express the fact that the directions oppose each other. Accordingly, in the case of UDP, the first packet is expressed by a positive number. The next packet is expressed by a positive number if the direction is the same as the first packet and by a negative number if the direction is the opposite.

To present the payload size and direction of the first N packets in a flow, we use a method to represent them as vectors. A flow is represented as f and the payload size of f's i-th packet is defined as $s_i(f)$. If τ is a function for representing a flow as an N-dimensional vector, it is defined as Eq. (1).

$$\tau(f) = (s_1(f), \ldots, s_N(f)) \tag{1}$$

This study is based on the distance between $\tau(f)$ for grouping and classifying flows. The distance between two flows, f and f', is calculated by using the city-block distance. If d is the distance between two vectors in the N dimension, it is represented as Eq. (2).

$$d(\tau(f), \tau(f')) = \sum_{i=1}^{N} |s_i(f) - s_i(f')| \tag{2}$$

Fig. 4 shows the process of generating statistic signatures. The flow preprocessing stage is where an operation is performed to purify the ground-truth flow data for delivering only normal application flows to the next module. The preprocessing module first removes all abnormal flows, such as a flow with no payload packets and a flow with missing packets. Next, the preprocessing module resolves the out-of-order and retransmission problems in TCP flows to collect only the normal flows. Each flow completing the preprocessing operation sequentially enters a grouping module input. Whenever flows sequentially enter the input, the grouping module first converts each flow into a flow vector ($\tau(f)$) for distance calculation. Afterwards, an operation is performed to find a group (minDistGroup) whose distance (d) is nearest to the flow. This operation serves to find a group with the nearest city-block distance from among the groups for which the flow matches every group attribute. If a minDistGroup is found, it updates the minDistGroup's centroid vector; if there is no minDistGroup, it creates a new group.

Group optimization is an operation employed for removing flows that belong to no group and for minimizing the initial distance threshold of each group. In other words, within a group, there should only be flows with the same characteristics, and an application should only be placed in a single group. Each group becomes a statistical signature of each process. This becomes a basis for traffic classification; however, a collision among groups generates a false positive (FP) in actual classification results. A collision means that a flow belongs to multiple groups. To solve this problem, the distance threshold of the group with a collision is lowered to remove the flow generating the collision. This method changes the FP into a false negative (FN). In other words, if the method cannot clearly decide if the flow belongs to an application, it does not classify it. In many cases, traffic misanalysing causes a greater problem than lack of traffic identification. In addition, when applying various methodologies to classify traffic—such as in the multi-level classification methodology that many traffic classification studies are currently using—the case in which there is no FP despite an FN occurring can be easily harmonized with other methodologies. When a group is finally completed, the group is used as a signature. This algorithm finally creates statistic signatures in an XML file for each application.

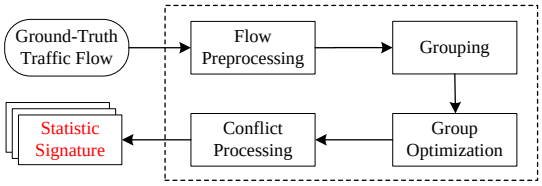

Fig. 4. Statistic Signature Generation

A payload signature is defined as the sequence of invariable bytes that appear in several initial packet payloads of the relevant application flow [12]. That is, the pattern of a payload signature is the sequence of fixed bytes that appear in a packet payload within the relevant application flow. In this paper, we define a payload signature as invariable byte sequences of the n-th packet within the initial ten-packet payloads of relevant application flows.

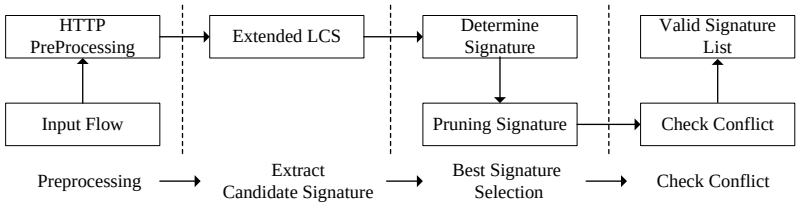

Fig. 5. Payload Signature Extraction Process

Fig. 5 depicts the payload signature extraction phase. The payload signature generation system produces a signature by an individual application unit. It transforms all traffic data generated by an application into the flows to use as input.

Basically, we use the longest common subsequence (LCS) algorithm [8] to extract payload signatures. To enhance the accuracy of our LCS-based payload signatures, we first enter the HTTP preprocessing phase. The HTTP preprocessing phase merely excludes HTTP packets. Because the HTTP protocol is used for transporting data by many applications, HTTP protocol keywords cannot be used for detecting applications. In addition, payload data transported by the HTTP protocol cannot be used for detecting applications. Therefore, to reduce load for LCS performance, and to generate accurate payload signatures, it performs an operation to remove the HTTP protocol keyword and payload data for HTTP traffic.

At the Extract Candidate Signature module, all potential candidate signatures are extracted by the LCS algorithm [8]. LCS finds the longest common substrings from two input strings. When multiple input strings ($\{f_1, f_2, \ldots, f_n\}$) of an application are given, there can be more than two of set S of the substrings ($S_1 = \{s_{1,1}, s_{1,2}, \ldots, s_{1.x}\}, S_2 = \{s_{2,1}, s_{2,2}, \ldots, s_{2,y}\}, \ldots$), for which the summation of all substring lengths of each set S is the same as L_{max}. This phase uses the LCS to find all possible substring sequences with maximum length (L_{max}) to pass them to the next phase. In other words, the signature extracted in this phase is selected through the process as given in Eq. (3).

$$\text{find all } S_i, \text{such that } L_{max} = |S_i| = \sum_{k=1}^{n} |s_{i,k}|, \; S_1 = \{s_{1,1}, s_{1,2}, \ldots, s_{1.x}\}, S_2 = \{s_{2,1}, s_{2,2}, \ldots, s_{2,y}\} \tag{3}$$

The Best Signature Selection module searches for the optimum signature from the candidate signatures. As shown in Eq. (4), the criterion that selects the best signature chooses the substring sequence ($S_k = \{s_{k,1}, s_{k,2}, \ldots, s_{k,z}\}$), which has the longest substring and lowest number of substrings, as the best signature from among the candidate substring sequences ($S_1 = \{s_{1,1}, s_{1,2}, \ldots, s_{1.x}\}, S_2 = \{s_{2,1}, s_{2,2}, \ldots, s_{2,y}\}, \ldots$).

$$\begin{gathered} S_k = \{s_{k,1}, s_{k,2}, \ldots, s_{k,z}\}, \; \text{Count}(S_k) = z, \text{such that } k \le \text{Count}(S_i) \text{for all } S_i, \\ \text{Max}(S_k) = \text{Max}\{s_{k,1}, s_{k,2}, \ldots, s_{k,z}\} = |s_{k,m}|, \text{such that } |s_{k,m}| \le |s_{i,j}| \text{ for all } s_{i,j} \text{ in all } S_i \end{gathered} \tag{4}$$

In addition, this stage is where substrings with a length below a certain threshold value from the substring sequence ($\mathbf{S_k = \{s_{k,1}, s_{k,2}, \ldots, s_{k,z}\}}$) and which were selected as the best signature were removed. This process serves to extract only substrings significant for detecting the application, as in Eq. (5).

$$\text{remove } s_{k,i} \text{ from } S_k \text{ if } (s_{k,i} == \text{text} \;\&\; |s_{k,i}| \le 2) \;\&\&\; \text{remove } s_{k,i} \text{ from } S_k \text{ if } (s_{k,i} == \text{binary} \;\&\&\; |s_{k,i}| \le 1) \tag{5}$$

At the Check Conflict phase, the selected signature is examined to determine if it conflicts with any other signatures in the list of existing effective signatures. Signatures are generated in application units and there is a risk that signatures generated from an application can overlap with ones generated from other applications. Therefore, to generate a unique signature for the

application, a duplication test should be performed to verify whether the signature does not overlap with that of other applications.

4.2 Classification Algorithms

In this section, we illustrate classification algorithms, which consist of one behavior-based algorithm and three signature-based algorithms with signatures that were explained in Section 4.1.

4.2.1 Signature-based Classification Algorithm

Fig. 6 presents a flow chart of the traffic classification algorithm that uses header, statistic, and payload signatures. If a packet enters the system, whether the flow of the relevant packet has been generated is checked. If the flow already exists, it is updated; if the flow does not exist, a new flow record is generated. If a flow has already been identified as an application, the next packet is inspected. If the application has not yet been determined, it is analyzed by comparing it with the signatures of the three (header, statistic, and payload) classification systems. The signature-based classification modules are activated every time packets in a flow are captured, making it possible to get the identification result before the completion of a flow.

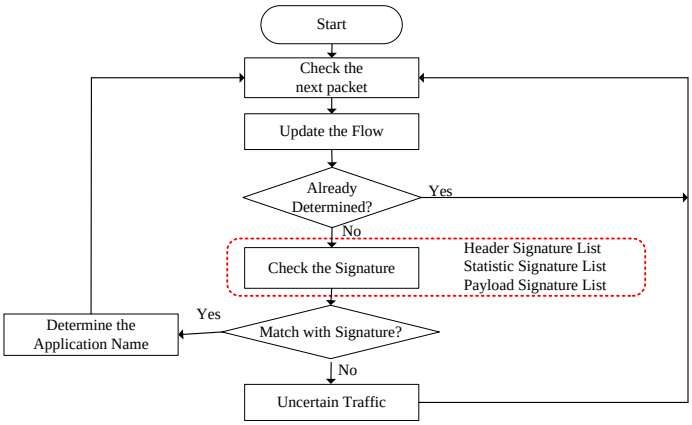

Fig. 6. Signature-based Classification Algorithm

Three types of signatures are described by different forms depending on their respective characteristics as shown in the equation below, and they are processed by independent classification modules. The header-signature- and statistic-signature-based classification modules construct a signature in a hash table structure. The payload-signature-based module constructs the signature with automata to improve the application identification speed and efficiency.

$$\text{Header Signatures} = \{(hs, name) |\ hs = \{IP, port, protocol\}, name = application\ name\}$$
$$\text{Statistic Signatures} = \{(\tau, d, name) |\ \tau = \{s_1, s_2, s_3, s_4, s_5\}, d = distance, name = application\ name\}$$
$$\text{Payload Signatures} = \{(re, name) |\ re = regular\ expression, name = application\ name\}$$

If the flow matches one of the three signatures, it is classified as a flow of the relevant application. The undetermined flow is respectively classified by the three signature-based classification modules. If the application is detected by more than one classifier and their detection results are the same, the flow's application is determined by the detected application. However, if the application detection results are different in more than two classifiers, the

flow's application is determined depending on the priority of the classification algorithm (BAC > PSC > SSC > HSC) at the FI.

It cannot be determined what application the previous packet is from; therefore, if the flow does not match any of the signatures, the next packet is inspected. Flows that have not been classified in the second-level signature-based systems and behavior-based classification system are classified using a correlation algorithm. This correlation algorithm is described in Section 4.3.

4.2.2 Behavior-based Classification Algorithm

Behavior-based classification is a method of categorizing applications that are difficult to classify by signature-based classification methods. In this method, they are classified by analyzing the specific behavioral patterns of each application. Many signature-based and machine-learning methods have been proposed for application-level traffic classification; however, the results of applying them to applications, such as Skype or uTorrent, show low reliability. In the behavior-based classification module, we gather ground-truth traffic of a target application, analyze the application's unique and distinctive traffic pattern, and construct a behavior-based analysis model to identify the application traffic. Currently, we have constructed a Skype traffic identification model that provides good performance.

It is difficult to apply a signature-based classification system to Skype because the port used by Skype randomly changes and the data transmitted on the network is encrypted [22][23]. The keys to the Skype traffic detection algorithm are meant to detect Skype traffic through deep packet inspection (DPI) of the several initial packets of traffic flow, and to detect Skype flows generated from other Skype clients (SCs) by building on this basis a list of hosts {IP, port} where SCs are installed.

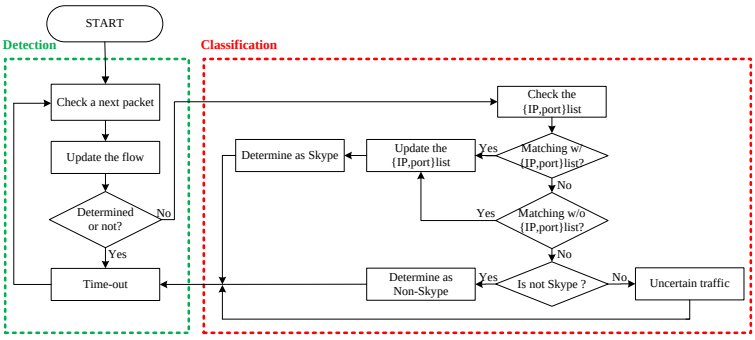

Fig. 7. Behavior-based Detection Algorithm for Skype

Fig. 7 provides a flow chart of the Skype application traffic detection algorithm. First, when a packet is captured, new flow information is generated based on the relevant packet; in the case of an already-generated flow, the flow information is updated. If it has been already determined whether the flow is from Skype, no further inspection is conducted. However, if that determination has not yet been made for a particular flow, the behavioral classification phase begins. In the classification phase, a determination is made whether the relevant flow is from Skype, or a judgment is made that more packets should be inspected (uncertain traffic). Additionally, if the relevant flow is determined to be a Skype flow, the information of the host {IP, port} is added to the host list.

At present, we are analyzing only Skype traffic by behavior-based classification methods. To date, the uTorrent P2P application is more than 90% classified by using the payload-signature method. However, uTorrent adopts a bypass mechanism by using UDP to

avoid firewall and traffic engineering systems. Therefore, an additional behavior-based classification method is required for uTorrent traffic. We have begun considerable focus on uTorrent.

4.3 Correlation-based Classification

In this section, the correlation-based classification algorithm that is the core of ITCS is described. Signature-based classification methods previously proposed pose the precondition that the signature of the relevant application must be known. Therefore, the problem is that whenever a new application appears, the signature of the relevant application must be extracted. Additionally, there is application traffic for which a signature cannot be extracted, even by our various methods; that is, there is application traffic with no possible signature. Of the traffic that cannot be classified by signature, however, some traffic can be classified by using correlations among flows.

Correlation-based traffic analysis is a method of expressing correlations among traffic flows using weighted values based on certain characteristics, such as the 3-tuple level address system (IP address, port number, and transport protocol) of Internet traffic, the traffic generation time, and the generation form. In addition, the method classifies traffic into application programs by applying the threshold of the weighted value. Using this method, diverse correlations among flows analyzed by existing signature-based classification methods can be identified, and our analysis completeness can be enhanced using these correlations. In this paper, the three following traffic correlations have been identified and applied.

(1) Server-Client-based Correlation

The server-client-based correlation methodology is based on the assumption that a server 3-tuple (IP address, port number, and L4 protocol) is used by only one application. Of the traffic analyzed by the preceding signature-based classification system, if there is a classification result that includes a server, an assumption is made that the 3-tuple of the relevant server has been used by the relevant application, and the traffic of the client that communicates with the relevant server in the network being analyzed is classified as that application.

(2) Generation-Time-based Correlation

The generation-time-based correlation methodology begins with the assumption of a high possibility that traffic generated within a certain period by only one host is from the same application. This method groups the traffic not classified by the preceding methods into hosts and short time intervals. If there is traffic classified among the relevant groups, it classifies all traffic that belongs to the group as belonging to the relevant application. This method requires a preceding experiment to establish a period adequate for the relevant network because the completeness and accuracy of the analysis are affected by the period used in the grouping standard.

(3) Host-Host-based Correlation

The host-host-based correlation methodology begins with the assumption of a high possibility that the communication occurring between two hosts is generated by one application. Therefore, if only part of the communication between two hosts has been classified by the preceding methods, we classify the unclassified traffic as being from an application of the classified traffic. In the case in which communications data is sent and received, the communications for control and those for data transmission occur at different ports. In this case, if the traffic that has transmitted the data has been classified, the traffic for control is

classified as the same application. The traffic between hosts that demonstrates this form is effectively analyzed using host-host correlations.

The priority of the above classification methods is server-client-, generation-time-, and then host-host-based correlations. The target of this correlation-based classification method is only the traffic not analyzed by the other methods. Accordingly, additional classification is made possible without decreasing the accuracy of the signature classification result by giving a high priority to the server-client-based correlation method, which has the highest accuracy.

The greatest advantage of this flow correlation-based classification algorithm is that it can classify, based on correlations, the traffic that could not be analyzed by the preceding signature-based classification system. A limitation of the signature-based classification method is that it cannot classify, or it incorrectly classifies, the traffic if it does not have a matching signature. However, when a flow correlation is used, the traffic without a signature can still be classified using the correlation among flows. Another advantage is that the proposed correlation methods can be applied in real-time manner, which makes it possible to get the classification result at the beginning of a flow.

Nevertheless, flow correlation has a limitation; that is, the performance of the flow correlation-based classification is highly dependent on the preceding classification results. A single wrong identified flow from a preceding system can cause a number of wrong results. In ITCS, the final product of this study, the signature-based classification method should provide highly accurate classification results to only the next module.

5. Evaluation and Verification

In this section, we describe the verification system and evaluation metrics to prove the validity of ITCS, and we discuss the result of applying the system to actual traffic on the campus network of Korea University.

5.1 Verification System and Evaluation Metrics

In this subsection, we present the environment built to verify the analysis of ITCS, as described in Section 4. All raw packets passing back and forth in a campus network were collected from the target link. **Fig. 8** shows the sites from which the traffic was collected. The Internet access point of the campus network was comprised of a router leading to the Internet and two core switches at the bottom. Traffic collection was performed through ITCS; an IPS and a device for QoS guarantee were connected to the router; and traffic was collected between two core switches under the device for QoS guarantee. The bandwidth of each link was 1 Giga.

ITCS collects packets and generates flows in one-minute units. It passes them to the classification system for analysis, which is completed within one minute. To be specific, ITCS is a near-real-time system because traffic is analyzed with a total delay of two minutes. It gathers minute unit classification results to show a classification result after one hour; it then gathers hour unit classification results to show day-of-the-week unit classification after 24 hours; finally, it gathers the results of a week to show the weekly classification result. These results are gathered for a month, and even for a year, with the data being used not only for real-time traffic classification, but also for trend analysis of application-level classification.

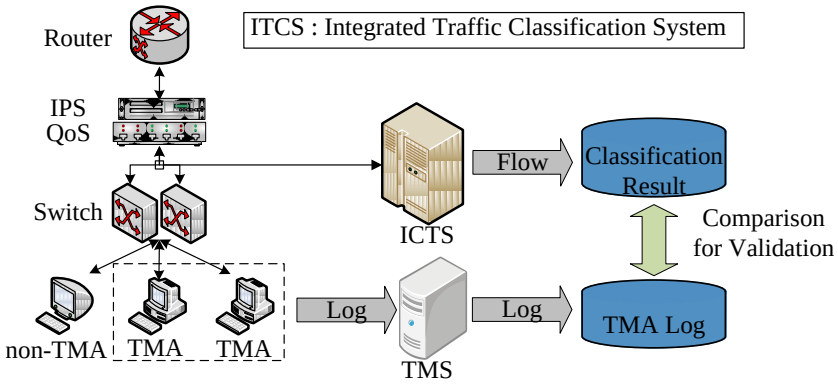

Fig. 8. System Deployment Environment

To verify the ITCS results, the following evaluation metrics were defined and used. The verification evaluation metrics were comprised of completeness, accuracy, overall performance, recall, and precision. Each evaluation metric was expressed in units of flow, packets, or bytes, respectively, for multilateral analysis. The first verification metric, completeness, is the amount of results classified by the relevant classification system expressed as a ratio of the whole traffic. Accuracy is expressed as the ratio of precise classification found by verifying how accurate the results classified by the relevant system were using GT. The last evaluation element, overall performance, is the ratio of the accurately classified portion of the whole classified traffic, and it is a product of completeness and accuracy. The classification accuracy can be divided into overall accuracy and individual application accuracy. It uses recall and precision to evaluate individual application classification accuracy.

5.2. Verification Result

To perform the integrated traffic analysis proposed in this study, we conducted a test for traffic on a campus network. For the test, classification results were derived based on all traffic generated on the campus network for six days using a general-purpose computer equipped with an Intel Core i7-2600 processor, a 3.4 GHz CPU, and 8 GB RAM.

To evaluate ITCS performance, a test was conducted for the traffic trace described in **Table 1**. The Flow column shows the number of the in/out flows of each day, the Packet column presents the number of packets, and the Byte column lists the number of bytes of the in/out traffic. First, three signatures (header, statistic, and payload) and a behavior algorithm for Skype were extracted with the traffic traces from Day 1 to Day 5. Application classification was then simultaneously performed in real-time for the Day 6 traffic.

Table 1. Test Traffic Trace

	Flows(10^3)	Packets(10^6)	Bytes(10^9)
Day1	3,089 / 48,872	111 / 1,801	86 / 1,541
Day2	3,200 / 21,767	95 / 749	67 / 624
Day3	3,352 / 55,940	90 / 2,034	65 / 1,707
Day4	3,450 / 53,353	136 / 1,969	112 / 1,673
Day5	2,932 / 52,282	95 / 6,051	70 / 58,334
Day6	3,232 / 50,322	116 / 1,852	106 / 1,547

Fig. 9 depicts the ITCS classification results. The figure shows the classification results of data from the sixth day. The applications are listed by name and the classification results are shown by flow in a pie chart. The chart presents an application-level classification of the 11 most frequently used applications, while other applications are grouped in the "etc." and "unknown" categories. The highest percentage of the traffic was generated by uTorrent, one of the torrent applications and occupying about 50% of total campus traffic. This is the most popular torrent application in Korea among torrent applications such as BitTorrent, VUZE, and libTorrent. More detailed analysis on the torrent traffic is necessary to understand the behavior and pattern of the traffic as well as the identity of the traffic, which we are planning to study in our future work.

We tried to compare our classification result with the other system, such as Bro [24]. Since the traffic classes covered by the systems are different from each other, it was difficult to compare the result one by one. However, we could certify that the analysis result on application-protocol level showed the same proportions of HTTP, FTP, SMTP, SSH, etc. When we applied our signatures to the Bro, it showed that some of the common applications are classified with the same rate as uTorrent, pdpop, nateon and ndrive, as shown in **Fig. 9**.

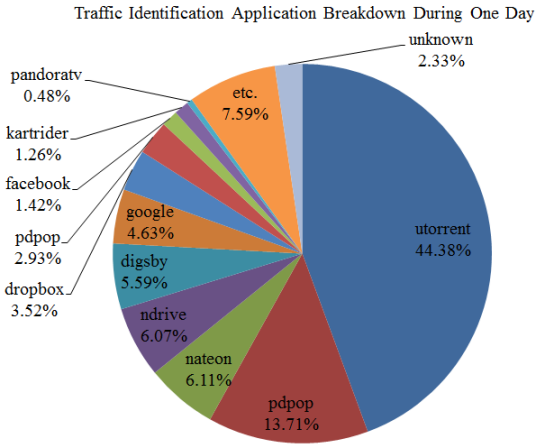

Fig. 9. Application-level Classification Result

Table 2 shows ITCS classification completeness, accuracy, and overall performance. By comparing the ITCS classification result with other signature-based classification results, it is evident that the completeness of the proposed methodology is relatively high. The weakness of the header and statistic signature-based classification methodologies with low completeness is resolved by the correlation-based classification. In other words, the correlation-based classification can analyze the traffic that has not been analyzed by the three signature-based classification methods and behavior algorithm. Based on correlations among flows, the results show better overall completeness.

As shown in **Table 2**, our classification system yields a high accuracy of more than 97%, which is higher than other existing analysis systems. However, we can see that the accuracy of our system is slightly lower than the results from the header-signature-based classification system. The accuracy is lower primarily because of conflicts among different applications. The result of analyzing the traffic involved in conflicts showed that, in many cases, an application used several application-level protocols. In particular, conflicts among several Web Disk applications, which use the same application-level protocol, were the main cause of

accuracy deterioration. Additionally, because a diverse range of programs provide Web-based services, conflicts among these Web-based applications and iExplorer caused further misclassifications.

Table 2. ITCS Completeness, Accuracy, and Overall Performance

	Completeness			Accuracy			Overall Performance		
	Flow	Packet	Byte	Flow	Packet	Byte	Flow	Packet	Byte
Integrated	97.63%	95.47%	95.69%	97.94%	97.15%	96.88%	95.62%	92.75%	92.70%
Header Signature	24.83%	5.15%	3.99%	99.91%	99.92%	99.91%	24.81%	5.15%	3.99%
Statistic Signature	18.99%	30.31%	31.52%	97.20%	91.89%	98.11%	18.46%	27.85%	30.92%
Payload Signature	86.06%	76.75%	75.72%	97.49%	94.18%	95.40%	83.90%	72.28%	72.24%

For our purposes, overall performance is the degree of accurately classified traffic among total traffic. That is, overall performance is a product of completeness and accuracy. The result of overall ITCS performance shows an accurate classification of 92% in bytes for applications, which is superior to other existing classification systems. For existing signature-based classification systems, while only payload-signature-based classification shows an accurate completeness of more than 72%, the other signature-based methods show an extremely low accurate completeness of approximately 20%. If the conflict problem described above is resolved, a higher accurate completeness can be achieved.

Table 3. ICTS Precision and Recall

	Precision			Recall		
	Flow	Packet	Byte	Flow	Packet	Byte
utorrent	97.06%	96.30%	96.11%	100%	100%	100%
nateon	96.42%	95.98%	96.41%	100%	100%	100%
ndrive	95.66%	99.91%	99.97%	95.42%	96.35%	95.23%
melon	97.05%	98.02%	98.36%	100%	100%	100%
digsby	100%	100%	100%	97.44%	92.17%	94.25%
kartrider	99.35%	98.49%	99.10%	100%	100%	100%
pdpop	93.55%	94.64%	96.51%	100%	100%	100%
dropbox	94.66%	93.98%	96.19%	100%	100%	100%
pandoratv	96.76%	95.06%	95.52%	100%	100%	100%
facebook	98.33%	98.63%	98.93%	95.00%	99.32%	99.94%

Table 3 presents the results of ITCS precision and recall of the ten most frequently used applications in the campus network. We collected 500 flows using TMA agents as ground-truth data. It is evident that for both precision and recall, ITCS shows a high accuracy of more than 95% for most applications. The relatively low recall value for Digsby relates to it being an application that integrates various messengers. An object is not classified by Digsby if each messenger uses its own protocol.

6. Conclusion and Future Work

To guarantee Internet QoS and provide data-centric services, the importance of traffic classification is greater than ever. In this paper, we proposed ITCS, a multi-level, integrated, near-real-time method of performing application-level Internet traffic classification. The proposed system combines and compensates several signature-based, behavior-based, and correlation-based methods in parallel and cascading steps to achieve a satisfactory classification performance in a real operational network. In addition, we implemented ITCS

and deployed it on our campus network to prove the validity and feasibility of the proposed multi-level integrated classification method. Using this system, we supplemented the limitations of signature-based classification by integrating the classification results from preceding modules into the final correlation-based classification modules. Deployed in our campus network, ITCS analyzed Internet traffic generated by approximately 246 applications, and it demonstrated a 97.63% completeness and 97.94% accuracy in flows.

From this study, we confirmed that conflicts among different classification algorithms reduce classification accuracy. Accordingly, we intend to conduct an in-depth study to resolve the conflicts among individual classification algorithms. Furthermore, we plan to build a solid real-time system possessing high reliability from the accuracy enhancement of individual signature-based classification algorithms and by additional studies on the real-time processing of flows on the basis of the current ITCS.

References

[1] Khorsandroo. Sajad, Rafidah Md Noor, and Sayid Khorsandroo, "A Generic Quantitative Relationship to Assess Interdependency of QoE and QoS," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 7, no. 2, pp. 327-346, February, 2013. Article (CrossRef Link)

[2] M. S. Kim, Y. J. Won and James Won-Ki Hong, "Application-Level Traffic Monitoring and an Analysis on IP Networks," *ETRI Journal*, vol. 27, no.1, pp. 22-42, February, 2005. Article (CrossRef Link)

[3] A. Dainotti , A. Pescape and K. Claffy, "Issues and future directions in traffic classification," *IEEE Network: The Magazine of Global Internetworking*, vol. 26, no. 1, pp. 35-40, February, 2012. Article (CrossRef Link)

[4] Bermolen, P., Mellia, M., Meo, M., Rossi D. and Valenti, S., "Abacus: Accurate behavioral classification of P2P-TV traffic," *Computer Networks*, vol. 55, no. 6, pp. 1394-1411, April, 2011. Article (CrossRef Link)

[5] W. Li, M Canini, AW Moore and R. Bolla, "Efficient application identification and the temporal and spatial stability of classification schema," *Computer Networks*, vol. 53, no. 6, pp. 790-809, April, 2009. Article (CrossRef Link)

[6] Aceto, G., Dainotti, A., de Donato, W. and Pescape, A., "PortLoad: taking the best of two worlds in traffic classification," in *Proc. of INFOCOM IEEE Conference on Computer Communications Workshops*, pp. 1-5, March 15-19, 2010. Article (CrossRef Link)

[7] J. S. Park, S. H. Yoon and M. S. Kim, "Software Architecture for a Lightweight Payload Signature-based Traffic Classification System," in *Proc. of Traffic Monitoring and Analysis Workshop*, vol. 6613, pp. 136-149, April 27. 2011. Article (CrossRef Link)

[8] Byung-Chul Park, Young J. Won, Myung-Sup Kim and James Won-Ki Hong, "Towards Automated Application Signature Generation for Traffic Identification," in *Proc. of IEEE/IFIP Network Operations and Management Symposium*, pp. 160-167, April 7-11, 2008. Article (CrossRef Link)

[9] Sung-Ho Yoon, Jin-Wan Park, Young-Seok Oh, Jun-Sang Park and Myung-Sup Kim, "Internet Application Traffic Classification Using Fixed IP-port," in *Proc. of Asia-Pacific Network Operations and Management Symposium*, LNCS5787, pp. 21-30, September 23-25, 2009. Article (CrossRef Link)

[10] IANA port number list, IANA, http://www.iana.org/assignments/port-numbers.

[11] G. Cheng and S. Wang, "Traffic classification based on port connection pattern," in *Proc. of International Conference on Computer Science and Service System*, pp.914 – 917, June 27-29, 2011. Article (CrossRef Link)

[12] Khalife, J. M., Hajjar, A. and Díaz-Verdejo, J., "Performance of OpenDPI in Identifying Sampled Network Traffic," *Journal of Networks*, vol. 8, no. 1, pp. 71-81, January, 2013. Article (CrossRef Link)

[13] Y. Jin, N. Duffield, J. Erman, P. Haffner, S. Sen, and Z.-L. Zhang, "A modular machine learning system for flow-level traffic classification in large networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 1-34, March, 2012. Article (CrossRef Link)
[14] Ying Zhang, Hongbo Wang and Shiduan Cheng, "A Method for Real-Time Peer-to-Peer Traffic Classification Based on C4. 5,"in *Proc. of IEEE International Conference on Communication Technology*, pp. 1192-1195, November 11-14, 2010. Article (CrossRef Link)
[15] Kuldeep Singh and Sunil Agrawal, "Comparative Analysis of five Machine Learning Algorithms for IP Traffic Classification," in *Proc. of Internation Conference on Emerging Trends in Networks and Computer Communications*, pp. 33-38, April 22-24, 2011. Article (CrossRef Link)
[16] Jaehak Yu, Hansung Lee, Younghee Im, Myung-Sup Kim and Daihee Park, "Real-time Classification of Internet Application Traffic using a Hierarchical Multi-class SVM," *Transactions on Internet and Information Systems*, vol. 4, no. 5, pp. 859-876, October. 2010. Article (CrossRef Link)
[17] Nen-Fu Huang, Gin-Yuan Jai and Han-Chieh Chao, "Early Identifying Application Traffic with Application Characteristics," in *Proc. of IEEE International Conference on Communications*, pp. 5788-5792, May 19-23, 2008. Article (CrossRef Link)
[18] Rentao Gu, Minhuo Hong, Hongxiang Wang and Yuefeng Ji, "Fast Traffic Classification in High Speed Networks," in *Proc. of Asia-Pacific Network Operations and Management Symposium*, LNCS 5297, pp. 429–432, October 22-24. 2008. Article (CrossRef Link)
[19] Z. Zhou, T. Song and F. Wenliang, "RocketTC: a high throughput traffic classification architecture," in *Proc. of IEEE Computing Network and Communications*, pp. 407-411, February 2. 2012. Article (CrossRef Link)
[20] Xie, G., Iliofotou, M., Keralapura, R., Faloutsos, M. and Nucci, A., "SubFlow: towards practical flow-level traffic classification," in *Proc. of IEEE INFOCOM*, pp. 2541-2545, March 25-30, 2012. Article (CrossRef Link)
[21] L. Bernaille, R. Teixeira, and I. Akodkenou, A. Soule and K. Salamatian, "Traffic Classification on the Fly," *ACM SIGCIMM Computer Communication Review*, vol. 36, no. 2, pp. 23-26, April. 2006. Article (CrossRef Link)
[22] Adami D, Callegari C, Giordano S, Pagano M. and Pepe T. "Skype-hunter: a real-time system for the detection and classification of skype traffic," *International Journal of Communication Systems,* vol. 25, no. 3, pp. 386–403, February, 2011. Article (CrossRef Link)
[23] Marcell Perenyi, Andras Gefferth, Trang Dinh Dang and Sandor Molnar, "Skype traffic identification," in *Proc. of IEEE GLOBECOM*, pp. 399-404, November 26-30, 2007. Article (CrossRef Link)
[24] Bro, http://bro-ids.org/index.html.

Mi-Jung Choi is an associate professor in the Department of Computer Science, Kangwon National University, Korea. She received her B.S. degree in CS from Ewha Womans University in 1998, and M.S. and Ph.D. degrees from the Dept. of CSE at POSTECH in 2000 and 2004, respectively. She was a Post-doc fellow at INRIA, France from Oct. 2004 to Sep. 2005 and at School of Computer Science, University of Waterloo, Canada from Nov. 2005 to Oct. 2006. Her research interests include traffic measurement, M2M network and service management, and mobile abnormality detection and prediction.

Jun-Sang Park received the B.S. and M.S. degree in computer science from Korea University, Korea, in 2008 and 2010, respectively. He is currently a Ph.D. candidate student of Korea University, Korea. His research interests include Internet traffic classification and network management.

Myung-Sup Kim He received his B.S., M.S., and Ph.D. degree in Computer Science and Engineering from POSTECH, Korea, in 1998, 2000, and 2004, respectively. From September 2004 to August 2006 he was a postdoctoral fellow in the Department of Electrical and Computer Engineering, University of Toronto, Canada. He joined Korea University, Korea, in 2006, where he is working currently as an associate professor in the Department of Computer and Information Science. His research interests include Internet traffic monitoring and analysis, service and network management, and Internet security.