Application Traffic Classification using PSS Signature

Jae-Hyun Ham[1,2], Hyun-Min An[2] and Myung-Sup Kim[2]
[1] The 2nd R&D Institute-1
Agency for Defense Development, Korea
[2] Dept. of Computer and Information Science
Korea University, Korea
[e-mail: {jhham, queen26, tmskim}@korea.ac.kr]
*Corresponding author: Myung-Sup Kim

Received November 26, 2013; revised April 10, 2014; accepted May 20, 2014; published July 29, 2014

Abstract

Recently, network traffic has become more complex and diverse due to the emergence of new applications and services. Therefore, the importance of application-level traffic classification is increasing rapidly, and it has become a very popular research area. Although a lot of methods for traffic classification have been introduced in literature, they have some limitations to achieve an acceptable level of performance in real-time application-level traffic classification. In this paper, we propose a novel application-level traffic classification method using payload size sequence (PSS) signature. The proposed method generates unique PSS signatures for each application using packet order, direction and payload size of the first N packets in a flow, and uses them to classify application traffic. The evaluation shows that this method can classify application traffic easily and quickly with high accuracy rates, over 99.97%. Furthermore, the method can also classify application traffic that uses the same application protocol or is encrypted.

Keywords: Application-level Traffic Classification, Application Identification, Statistical Signature, Signature-based Classification, Statistics-based Classification

This research was supported by a Korea University Grant.

http://dx.doi.org/10.3837/tiis.2014.07.004

1. 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 is not related to the purpose of the organization, such as P2P and game traffic. For these policies, fast and accurate traffic classification in the application layer is essential. Application-level traffic classification is a process that collects network packets and determines the identity of the application [1]. Accurate real-time application-level traffic classification is an important part of determining the reliability of monitoring and controlling the application traffic of individual applications [1][2][3].

Due to the importance of application-level traffic classification, a lot of methods have been introduced. Traditional methods commonly involve port-based or payload-based classification [4]. They were effective and precise in the early days of the Internet. However, the reliability of port-based classification is currently diminishing, because some users use inconsistent ports intentionally. At the same time, payload-based classification also has to face limitations in complexity, privacy issues and encrypted applications, which makes it difficult to classify application traffic effectively.

Recently, several methods have been introduced that use statistical flow information to address the limitations of port-based and payload-based classification [5][6]. The classification approach that uses statistical information has some advantages. For example, it works for encrypted traffic, the usage of which has been increasing recently. In addition, it does not need to analyze the payload information of the packets, so it can classify the traffic quickly. A problem to this approach is that it has to wait until the end of the flow to complete the statistical information, so it cannot classify the traffic in real time. To overcome this problem, methods that use the first N packets of a flow have been studied, but they are also difficult to apply to real-time traffic classification in high-speed real operation networks. They require computation cost for the feature extraction to produce the statistical information, and because their machine learning (ML) algorithms have very high computation complexity, those methods cannot be applied to real-time classification in high-speed real operation networks. Their results are also not categorized by each application, because they classify the application traffic into application protocols. They classify several applications into one application protocol when the applications use the same protocol. Therefore, they cannot be applied to many network management and operation policies that adjust each application.

In this paper, we propose a novel application-level traffic classification method using payload size sequence (PSS) signature. PSS signature represents the unique flow pattern of each application that can be utilized to distinguish applications. Our method generates PSS signatures for each application with statistical information of flows from application traffic traces. And, it classifies application traffic easily and quickly in real operation networks through simple signature matching of new flows to the PSS signatures of each application.

In the generation of PSS signatures for an application, flows of its traffic traces are converted into PSS vectors using packet order, direction and payload size of the first N packets of each flow. PSS vectors are grouped according to similarity between PSS vectors by our flow grouping algorithm to identify unique flow patterns of the application. The groups of PSS vectors are optimized and the PSS signatures of the application are extracted from each group by our group optimization and signature generation algorithm, respectively. For classification of application traffic, the PSS vector of a new flow in a real operation network is compared

with each PSS signature using our signature matching algorithm to determine the application of the flow.

The proposed method has three advantages. First, it can classify the application traffic in real time and can apply well to high-speed real operation networks that deal with a large amount of traffic. It uses only the packet order, direction and payload size of the first N packets of a flow during traffic classification, so it does not need to analyze packet payload data and can make PSS vectors without computation cost for the feature extraction from the flow. It also uses simple comparison for PSS signature matching that has extremely low computation complexity in comparison to ML algorithms. As a result, our method can operate effectively in real operation networks. Second, it can obtain highly accurate classification results, because it uses PSS signatures that reflect unique flow patterns of each application. Our evaluation shows that it can classify application traffic with high accuracy rates of more than 99.97%. Third, it can classify application traffic into each application, not application protocol, because it uses unique PSS signatures per application. Our evaluation also shows that it can classify application traffic that uses the same application protocol into each application.

The remainder of this paper is organized as follows. Section 2 briefly reviews and summarizes the previous work. Section 3 introduces our proposed method in detail. Section 4 describes our experimental method and analyzes classification results. The conclusion and future work are given in Section 5.

2. Related Work

A lot of traffic classification methods that use statistical information of application traffic flows have recently been studied [5][6]. These methods commonly use ML algorithms with the features (port number, flow duration, inter-arrival time, packet size, etc.) that can be characteristics of the application traffic. Because they do not analyze the payload data, there are no privacy problems and they can classify the traffic faster than payload-based methods. They can also classify encrypted traffic. Furthermore, by using high-quality algorithms that are qualified in the field of ML, they can classify the traffic with highly accurate results.

Table 1 compares the recent ML-based traffic classification methods that use features as statistical information of the traffic flows. The “Feature Extraction Range” presents the range in a flow that is necessary to extract its features for traffic classification. In other words, it represents the amount of packets that need to be investigated in order to complete the features. “Full flow” means that the method requires flow completion to extract its features, and the method can extract the features at the end of the flow, so it cannot classify application traffic in real time. “Partial flow” means that the method uses part of a flow to extract its features. The “Feature Computation Cost” presents the computation overhead for the feature extraction. “Low” means that the method does not need the computation for the feature extraction, and “Average” indicates the method needs simple computation for extracting features and the number of the used features is lower than 10. “High” means that the method uses the features that require complex computation to be extracted, or uses more than 10 features that require simple computation. The “ML Algorithm” presents the machine learning algorithm used in the method, and it infers the computation complexity for traffic classification. Finally, the “Classification Traffic Class” presents application traffic class that the method classifies into. “Protocol” indicates the application protocol, and “Application” means each individual application. For example, “N Protocols, two applications” means that the method can classify application traffic into N application protocols and two individual applications.

Table 1. Comparison of Recent ML-based Traffic Classification using Statistical Flow Information

Related Work	Feature Extraction Range	Feature Computation Cost	ML Algorithm	Classification Traffic Class
Bernaille et al. [7]	Partial flow	Low	K-Means, GMM, HMM	N Protocols, two applications
Bujlow et al. [8]	Partial flow	High	C5.0	N Protocols, three applications
Jin et al. [9]	Full flow	Medium	Modular architecture (Three linear ML Algorithms)	N Protocols
Tan et al. [10]	Full flow	Medium	SVM optimized by Particle Swarm Optimization	N Protocols, one application
Yuan et al. [11]	Full flow	Medium	SVM	N Protocols
Runyuan et al. [12]	Full flow	High	Probabilistic Neural Networks	N Applications
Yin et al. [13]	Partial flow	Low	HMM	N Protocols, five applications

Table 1 shows that some methods extract features at the end of a flow [9][10][11][12], so they cannot apply to the real-time traffic classification because they can determine the application of a new flow after it finishes. To overcome this limitation, the methods that extract features in the first N packets of a flow are studied [7][8][13]. However, high feature computation cost or high computation complexity of the ML algorithm in the methods makes it difficult to achieve the real-time traffic classification in high-speed real operation networks. In addition, most of the methods mainly classify application traffic into each application protocol, not each application, which is not sufficient to be applied to network management or operation policies that adjust or control individual application traffic.

In order to overcome the limitations of previous methods using statistical information, the proposed method generates PSS signatures for each application in advance. During traffic classification, it composes PSS vectors from the first N packets of new flows without computation cost, and performs PSS signature matching with linear computation complexity for application traffic classification. It classifies the application traffic into each individual application and basically has advantages of traffic classification using statistical information.

3. Traffic Classification Methodology using PSS Signature

In this section, we describe the proposed traffic classification method after defining PSS and describing PSS characteristics to distinguish application traffic.

In order to classify application traffic, PSS signatures that are unique to each application are necessary. We define PSS signature mathematically and propose PSS signature generation method. After that, we describe the traffic classification method based on the PSS signatures. In this paper, PSS signatures are generated per individual application instead of application protocol. It can be utilized in a broad range of fields in comparison with previous methods that classify traffic into application protocol.

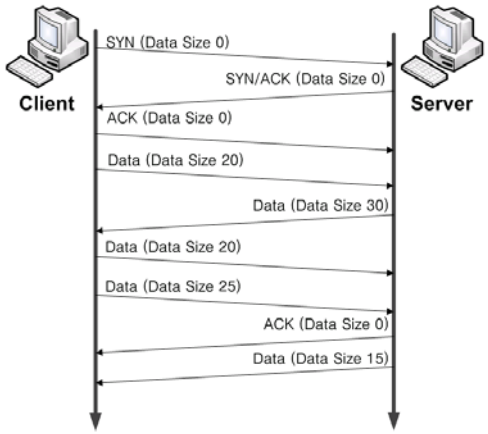
3.1 Payload Size Sequence (PSS)

In general, the first few packets of a flow communicate on the basis of pre-defined rules by an application. The first N packets of a flow can be used as a distinguishable feature to identify the application because they communicate according to pre-defined rules and are very different in each application [7].

In this study, we define the flow as a sequence set of packets transmitted in both directions based on a 5-tuple (source IP, destination IP, source port, destination port, and L4 protocol). Packet order is formed according to the collecting time.

The PSS of a flow is presented as a sequence set that is composed of the payload size and direction of the first internal N packets based on packet order. The payload size and transmission direction of each packet are expressed as an integer and "+/−", respectively. In the case of TCP, the transmission direction from client to server is defined as "+", and the opposite direction is defined as "−". In the case of UDP, because the distinction between client and server is not clear, the direction of the first packet is expressed as "+" and the opposite direction is determined as "−". The PSS is composed of only packets in which payload exists. The control packets such as SYN or ACK in TCP sessions are excluded. This prevents irregular control packets from affecting the PSS.

For example, if a flow communicates in both directions as shown in **Fig. 1**, the PSS of the flow has the values +20, −30, +20, +25, and −15, excluding the control packets such as SYN, SYN/ACK, and ACK.



$$PSS = (+20, -30, +20, +25, -15)$$

Fig. 1. Payload Size Sequence (PSS) of a Flow

From the observations of popular applications in the Korea University campus network, we found that applications can be distinguished with their PSSs. **Fig. 2** presents each PSS of four different applications, Dropbox, Microsoft Outlook, PuTTY, and Xshell. The PSSs that have the same packet order and direction are expressed as a flow group which is plotted with a polygonal line. The vertical axis shows the multiplication of the payload size and the direction of the packets. The horizontal axis shows packet order. The multiplication of the payload size and direction of the packet can have values from −1,460 to 1,460.

Most of applications do not have flows that communicate with one identical pattern of PSS such as PuTTY. The flows can be either divided with two or three specific patterns such as Xshell and Dropbox, or divided with a number of specific patterns such as Outlook. However,

PSSs of each application show their own unique characteristics when they are compared with other applications.

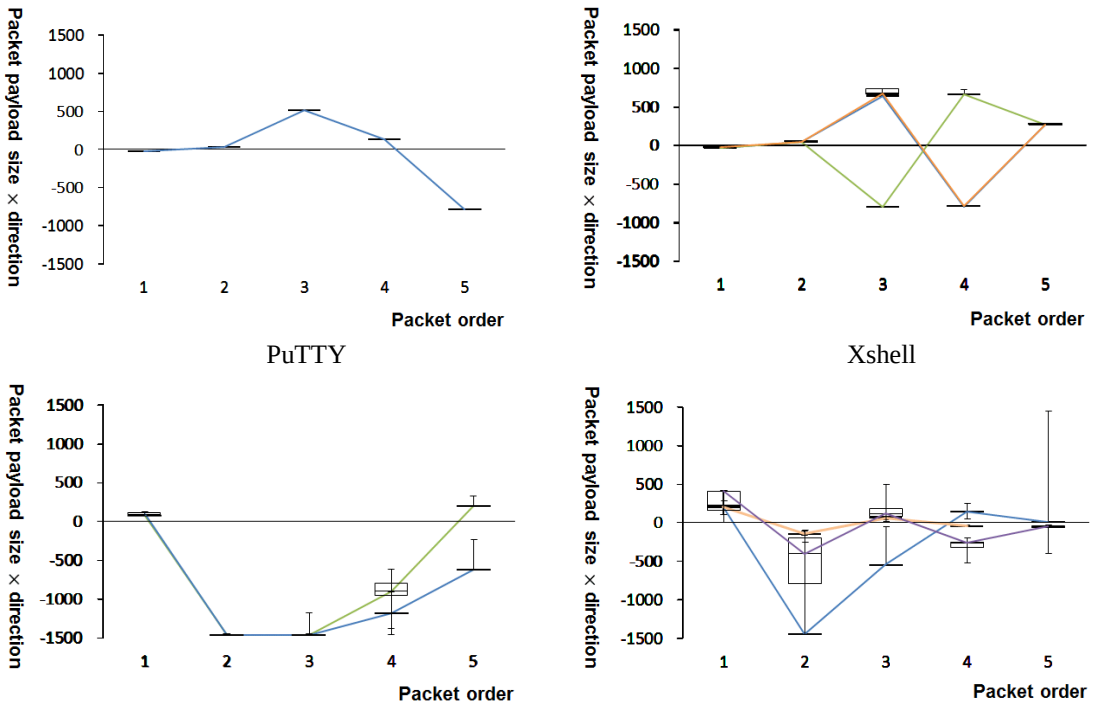


Fig. 2. The PSSs of Four Applications

A flow group indicates a specific flow pattern or application behavior. As shown in **Fig. 2**, flows of each application have regular flow patterns. The regular flow patterns can distinguish between different applications. Therefore, we can know that PSSs can be used to classify application traffic. Furthermore, distinguishable PSSs between PuTTY and Xshell which use the same SSH protocol indicate that PSSs can be used to classify application traffic which uses the same application protocol or is encrypted.

3.2 PSS Signature

A PSS signature represents each application's unique flow pattern that can be utilized to classify application traffic. The proposed method vectorizes flows from application traffic traces into PSS vectors per application, and groups PSS vectors on the basis of the similarity between PSS vectors to identify flow patterns of the application. In sequence, a PSS signature is generated with the combination of a representative PSS vector which represents PSS vectors of each group, and a distance threshold vector which includes PSS vectors of each group. If all of the individual PSS vectors are used as signatures, the number of signatures increases. As a result, managing the signatures becomes difficult and the system is overloaded for the identification of application traffic. Therefore, an optimal PSS signature combines the representative PSS vector and the distance threshold vector that represent each group by grouping the application PSS vectors.

The PSS vector v_k of a flow f_k can be expressed as (1), and each element $v_{k,i}$ of v_k can be written as (2). $d_{k,i}$ is the transmission direction of the *i*-th packet of f_k, and its value is either +1 or −1. $s_{k,i}$ is the payload size of the *i*-th packet of f_k.

$$v_k = (v_{k,1}, v_{k,2}, \dots, v_{k,n}) \tag{1}$$

$$v_{k,i} = \mathrm{d}_{k,i} \times \mathrm{s}_{k,i} \tag{2}$$

The similarity between two PSS vectors is expressed as the distance vector that has elements as distance between two PSS vectors by each dimension. Similarity is used in flow grouping and signature matching algorithms. The distance vector $d_{x,y} = d(v_x, v_y)$ between v_x and v_y can be written as (3).

$$d(v_x, v_y) = (|v_{x,1} - v_{y,1}|, |v_{x,2} - v_{y,2}|, \dots, |v_{x,n} - v_{y,n}|) \tag{3}$$

PSS signature *s* is represented with the representative PSS vector and the distance threshold vector, as in (4). *c* is the centroid PSS vector of the PSS vector group, and *t* is the distance threshold vector, which can include every PSS vector of the PSS vector group. *s* is the combination of *c* and *t* of the PSS vector group.

$$s = (c, t) \tag{4}$$

Each PSS vector *v* of PSS vector group *V*(*s*), which is represented by the PSS signature *s* = (*c*, *t*), should satisfy (5). The similarity between all *v* that belong to *V*(*s*) and the representative PSS vector *c* should be less than or equal to distance threshold vector *t*.

$$d(v, c) \leq t \ \ for \ \forall v \in V(s) \tag{5}$$

Multiple PSS signatures of each application can exist and (6) presents PSS signature set *S*.

$$S = \{s_1, s_{,2}, \dots, s_n\} \tag{6}$$

PSS vectors of each application are extracted from flows of ground-truth traffic traces that are collected in advance. After that, the PSS signature set is generated on the basis of the PSS vectors. Application traffic can be classified according to the PSS signature set.

$$\hat{S} \ \text{ such that } \begin{cases} \text{minimize } |S| \\ \text{minimize } \sum t(s) \\ \text{maximize } \sum |V(s)| \end{cases} \tag{7}$$

In order to obtain an optimal PSS signature set $\hat{S}$ from the given ground-truth flows of each application, the condition of (7) should be satisfied. $|S|$ indicates the number of PSS signatures s of S, $t(s)$ is the distance threshold vector of PSS signature s, and $|V(s)|$ is the number of

PSS vectors of set $V(s)$ represented by PSS signature s. Thus, (7) indicates the minimization of the number of PSS signatures, the minimization of the sum of distance threshold vectors, and the maximization of the number of PSS vectors that belong to each PSS signature for obtaining an optimal PSS signature set.

Fig. 3 presents the flowchart of the proposed application traffic classification method using PSS signature. The method is divided into two stages: signature generation and traffic identification. In the signature generation stage, application flows of ground-truth traffic traces are vectorized to PSS vectors using (1) and (2). Next, the flows are grouped on the basis of similarity between two PSS vectors which is defined by (3). Finally, PSS signatures are extracted after optimizing the groups. In the traffic identification stage, a new flow is generated from a series of packets in a real operation network and is vectorized to a PSS vector using the first N packets. The PSS vector is classified into each application by signature matching with each PSS signature.

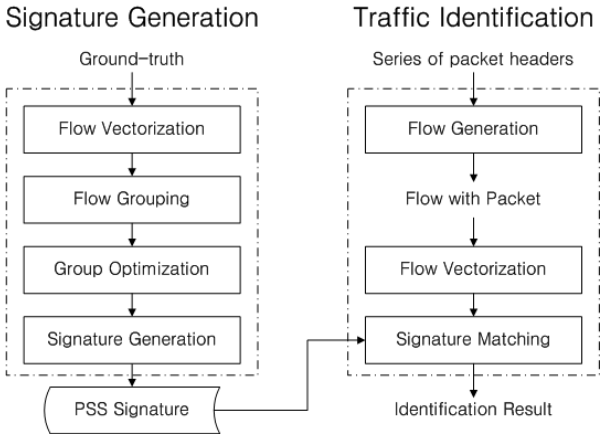


Fig. 3. Flow Chart of Application Traffic Classification using PSS Signature

3.3 PSS Signature Generation

In this section, we present some requirements of the ground truth traffic to generate optimal PSS signatures. Next, we describe PSS signature generation in detail. PSS signature generation is composed of flow grouping, group optimization and signature generation algorithms. The flow grouping algorithm is used to identify unique flow patterns for each application. The group optimization algorithm adjusts each flow group from the flow grouping to make an optimal PSS signature set, and the signature generation algorithm extracts PSS signatures from final groups.

3.3.1 Ground Truth Traffic

Ground truth traffic has some requirements to generate optimal PSS signatures. First, all types of traffic of the concerned applications must be included. Applications can perform several functions and generate slightly different traffic patterns. Therefore, all types of traffic generated from an application should be collected to extract accurate signatures. Second, the traffic of each application should have more than a certain number of flows. The signature that is generated with too few flows can cause a decrease of reliability and a decline of distinguishable traffic in the application classification. Third, the traffic of each application

needs to be collected from multiple hosts. If any application traffic is collected by one host, the characteristics of traffic can be dependent on that host. For example, many applications can change their port number in the settings menu, and the port number that is modified by a user can be used as a signature of that host but cannot be used in other hosts.

3.3.2 Flow Grouping

The application flows are divided into multiple flow groups by similarity of PSS vectors in order to extract PSS signatures of each application from the repeated and unique patterns of flows. Each flow group, as shown in **Table 2**, has six properties.

Table 2. Attributes of a Flow Group

Attribute	Description	Example
ID	Application Name	BitTorrent
Proto	L4 protocol: TCP/UDP	UDP
V	Set of Flow vectors	$\{v_1,v_2,v_3,\ldots,v_n\}$
Dim	Dimension of vector	5
C-vector	Centroid PSS vector	(+20,+30,-50,+20,-30)
T-vector	Distance threshold vector	(10, 10, 10, 10, 10)

In **Table 2**, "ID" is the identity of the application for the flow group. "Proto" is L4 protocol for the flow group, and it has the value of either TCP or UDP. The flow grouping is achieved by flows that have the same L4 protocol even if the flows are generated in the same application. "V" is the set of PSS vectors belonging to the group, and "Dim" is the dimension of the PSS vectors belonging to the group. All flows belonging to the same group have the same dimension. Because the dimension means the number of internal packets of the flow that is vectorized to the PSS vector, its value can be from 1 to N. However, the minimum value is defined as 3 in this study. This is because it is difficult to generate a sensible PSS signature with one- or two-dimensional PSS vectors. In addition, N is defined as 5 empirically with consideration for real-time and accurate traffic classification. Bernaille et al. [14] also shows that the first five packets of a flow are enough to distinguish each application. The C-vector is the centroid vector of the group and is used in the flow grouping task. The C-vector is calculated using all of the PSS vectors belonging to the group, and the calculation is repeated whenever a new PSS vector is added to or removed from the group during flow grouping. The T-vector is the condition of flow grouping and indicates the range of the group. The initial value of the T-vector must be a value that can include the largest number of flows that indicate the same flow pattern and can exclude flows that indicate different patterns. In this study, the initial value of every element of T-vector is defined as 10 on an experimental basis to identify unique flow patterns of each application properly in flow grouping.

The flows that are grouped into one group during flow grouping have the same ID, Proto, and Dim, and they satisfy (8). G_i indicates the i-th group, V_i is the set of PSS vectors belonging to G_i, and v_j means the j-th PSS vector of V_i. c_i and t_i are a C-vector and T-vector, respectively. Thus, (8) means that all of PSS vectors belonging to a specific group should be in the range that is formed with the C-vector and T-vector.

$$d(v_j, c_i) \leq t_i \text{ for } \forall v_j \in V_i, c_i, t_i \text{ of } G_i \tag{8}$$

Fig. 4 shows the pseudo-code of our flow grouping algorithm. Every flow of each application is entered into the flow grouping algorithm, after flows from ground-truth traces are represented as PSS vectors and divided according to each application. v_j is the j-th PSS vector. G_i means the i-th group, and c_i is the C-vector of G_i. t_i indicates the T-vector of G_i. $\tau_{i,j}$, which can have only 0 or 1, is newly defined for the flow grouping. When $v_j \in G_i$, it has the value of 1. Otherwise, it has the value of 0. The flow grouping is completed when $\tau_{i,j} = 1$ for all j.

input all PSS vectors of an application
output flow groups of an application

n, m : max number of groups, PSS vectors
i, j : index of group, PSS vector

Set $\tau_{i,j}$= 0, n = 0
for each of the PSS vector v_j, $0 \le j < m$
  Find a Group G_i , $0 \le i < n$ such that $d(v_j, c_i) \le t_i$
  if (G_i found) **then**
    $\tau_{i,j}$ = 1
  else
    n = n + 1 and i = n
    create a new group G_i and $\tau_{i,j} = 1$
  end if
  recalculate c_i through $c_i = \frac{1}{\sum_{j=1}^{m} \tau_{i,j}} \times \sum_{j=1}^{m} (\tau_{i,j} \times v_j)$
end for

Fig. 4. Pseudo-code of Our Flow Grouping Algorithm

The flow grouping algorithm performs the task after initialization of all $\tau_{i,j}$ values. It detects the group that can contain a PSS vector by comparing the T-vector and the distance vector between the PSS vector and the C-vector. If the detection succeeds, the PSS is assigned to the group and the C-vector of the group is recalculated. If the detection fails, the algorithm generates a new group and calculates the C-vector of the group. The flow grouping is completed when the task is performed for all PSS vectors.

3.3.3 Group Optimization and Signature Generation

Group optimization removes the inappropriate outlier PSS vectors and groups after completing the flow grouping. The outlier PSS vectors that are inside the group and do not meet the requirement of (8), are removed. In addition, the outlier groups that satisfy the condition of (9) are removed.

Because the C-vector of the flow group is continuously changing during the flow grouping, the outlier PSS vectors that do not belong to the final group appear, and they must be removed from the final group. In order to prevent the extraction of the meaningless behavior of application traffic as the PSS signature, and to minimize the number of PSS signatures of each application, the outlier groups that have less than a certain number of PSS vectors must be removed. (9) is the condition that removes the outlier groups. $|V_i|$ indicates the number of PSS

vectors belonging to the group G_i, and the *minimal flow count* is the pre-defined threshold value.

$$|V_i| = \sum_{j=1}^{m} \tau_{i,j} < minimal\ flow\ count \tag{9}$$

Next, a PSS signature is generated from each flow group. As shown in **Table 3**, each signature has five properties.

Table 3. Attributes of a PSS Signature

Attribute	Description	Example
ID	Application Name	BitTorrent
Proto	L4 protocol: TCP/UDP	UDP
Dim	Dimension of vector	5
C-vector	Centroid PSS vector	{+18,+31,-46,+15,-30}
T-vector	Distance threshold vector	{10, 8, 5, 7, 0}

The ID, Proto and Dim of the PSS signature are inherited from each of the flow groups. After that, our signature generation algorithm calculates the C-vector of the flow group using (10) and assigns it to the C-vector of the PSS signature. Because the group optimization algorithm removed the PSS vectors that did not meet the conditions in (8), the C-vector should be recalculated. The C-vector is used with the T-vector to identify the application of a new flow during the traffic classification.

$$c_i = \frac{1}{\sum_{j=1}^{m} \tau_{i,j}} \times \sum_{j=1}^{m} (\tau_{i,j} \times v_j)\ \ for\ \forall v_j \in V_i\ of\ G_i \tag{10}$$

In addition, the signature generation algorithm calculates the T-vector of the flow group and assigns it to the T-vector of the PSS signature. Each element of the T-vector of all flow groups has the same value as an initial value. However, the distance threshold of each group that can contain all flows is different, because each group has different density and distribution of PSS vectors after the flow grouping is completed. Therefore, the T-vector should be minimized in order to eliminate misclassification errors during traffic classification. The T-vector is calculated using (11), which means that the T-vector has elements as the maximum distance between each PSS vector and C-vector by each dimension. Each element of the T-vector cannot be more than 10, because we assigned its initial value as 10 for each group.

$$t_i = \{max(|v_{j,1} - c_{i,1}|), max(|v_{j,2} - c_{i,2}|), \ldots, max(|v_{j,n} - c_{i,n}|)\}\ \ for\ \ \forall v_j \in V_i, c_i\ of\ G_i \tag{11}$$

Fig. 5 shows the removal of outlier PSS vectors and the T-vector minimization on a two-dimensional space to facilitate understanding. As shown in the left figure, some PSS vectors do not belong to the final group. This phenomenon occurs because the centroid vector changes during the flow grouping. The outlier PSS vectors are removed to eliminate the misclassification error. The right figure of **Fig. 5** shows the minimization of the initial element value of the T-vector to the maximum distance between each PSS vector and C-vector by each dimension after the flow grouping. The final rectangle shape that is formed with the C-vector

and T-vector presents the geometrical form of the PSS signature of the application for the flow group on a two-dimensional space.

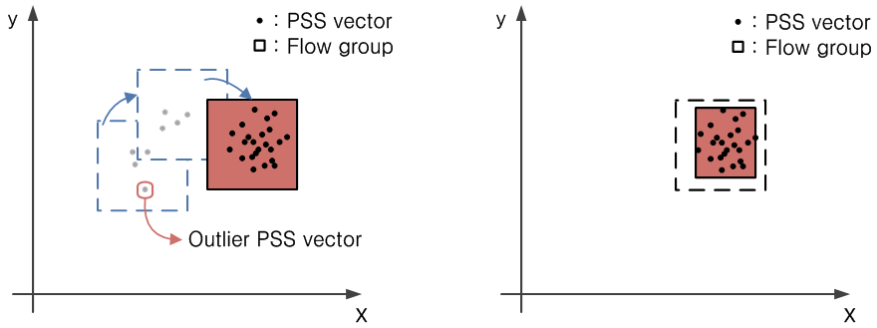


Fig. 5. Removal of Outlier PSS vectors and the T-vector Minimization on Two-Dimensional Space

Fig. 6 shows the pseudo-code of our group optimization and signature generation algorithms. They optimize the groups of all applications and generate the PSS signatures for each application from the groups. The optimized groups of each application indicate unique flow patterns that can identify new flows of the application effectively. Therefore, PSS signatures from the optimized groups can classify application traffic into each application with high accuracy.

input flow groups of all applications
output PSS signatures of application for flow groups

for each of the flow Group G_i
  Find v_j such that $\tau_{i,j} = 1$ and $d(v_j, c_i) > t_i$
  if (v_j found) **then** $\tau_{i,j} = 0$
end for
Find a Group G_i such that ($\sum_{j=1}^{m} \tau_{i,j} < minimal\ flow\ count$)
if (G_i found) **then** eliminate G_i and set $\tau_{i,j} = 0$ for all v_j of G_i where $\tau_{i,j} = 1$

for each of the flow Group G_i
  calculate c_i through $c_i = \frac{1}{\sum_{j=1}^{m} \tau_{i,j}} \times \sum_{j=1}^{m} (\tau_{i,j} \times v_j)$
  calculate t_i through each element k of $t_i = max(|v_{j,k} - c_{i,k}|)$ for each v_j
end for
Set PSS signature of application for the flow Group $G_i = (c_i, t_i)$

Fig. 6. Pseudo-code of Our Group Optimization and Signature Generation Algorithms

3.4 Traffic Identification

In the traffic classification, the traffic of a real operation network is captured and classified on the basis of the PSS signatures of each application. A new flow is generated with a series of packets that are captured from the real operation network. The flow is vectorized into a PSS vector using packet order, direction and payload size of its first N packets. The PSS vector is compared with each PSS signature to determine its identity. If the PSS vector of a new flow is

included in a PSS signature by the C-vector and T-vector, its application is determined as the ID of the PSS signature.

Thus, traffic identification finds the PSS signature s_i that presents in (12) for the PSS vector v of a new flow. At this time, s_i should have the same L4 protocol and dimension as v.

$$s_i \text{ such that } d(c_i, v) \leq t_i \text{ for } s_i = (c_i, t_i) \tag{12}$$

In this study, a PSS vector uses only packet order, direction and payload size of the first five packets of a flow, so PSS vectorization that converts a flow into a PSS vector with those features does not need any computation cost. In addition, PSS signature matching is simple similarity comparison operation with linear computation complexity as shown in (12). Therefore, traffic classification, by these simple PSS vectorization and signature matching algorithms, does not require high computation costs in comparison to previous methods, which enables real-time traffic classification in high-speed real operation networks.

input PSS vector v, all PSS signatures
output application of the PSS vector v

1: **Find** PSS signature s_i such that $d(c_i, v) \leq t_i$
2: **if** (one s_i found) **then** application of v = ID of s_i
3: **else if** (more than two s_i found) and (all s_i have the same ID) **then** application of v = ID of s_i
4: **else** application of v = *unknown*
5: **end if**

Fig. 7. Pseudo-code of Our Signature Matching Algorithm

Fig. 7 shows the pseudo-code of our signature matching algorithm. When the PSS vector of a new flow is entered, it looks for all PSS signatures that present in (12). If there is only one discovered signature, a new flow is classified as its ID. If the discovered signatures are more than two and have the same ID, a new flow is classified as their ID. However, a new flow will be classified as *unknown* if several PSS signatures with different IDs are found, which we define as the PSS signature conflict. In addition, the flow will be classified as *unknown* if no PSS signature exists that presents in (12).

4. Evaluation

In this section, we describe the result of traffic classification test in the campus network in order to verify the proposed application traffic classification using PSS signature.

4.1 Ground Truth Traffic

For our evaluation, we collected bi-directional packet traces from the Korea University campus network. The campus network was configured with one router at the Internet junction, so we collected the traffic traces from the router using port mirroring.

In order to evaluate traffic classification, it is crucial to obtain a firm ground truth. We deployed traffic measurement agents (TMAs) on selected hosts in the campus network and created the ground-truth traffic [15][16]. The ground truth traffic through those agents is more

reliable than using the result of the particular classification method to evaluate the classification method [17].

Table 4 presents the summary of the ground truth traffic we obtained for evaluation. The traffic is arranged by each application to verify that our method can classify the traffic into each application. Because the traffic is collected over a period of time, the traffic volume is various according to the frequency of use of each application and the amount of generated traffic.

Table 4. Ground Truth Traffic

Traffic Class	Description of Applications	Flow (10^3)	Packet (10^3)	Byte (10^6)
Skype	P2P communications	2.9	43.9	17.6
GomTV	Internet TV service	15.2	2,637.0	2,515.6
Naverlive	Video streaming	2.6	50,807.5	41,404.3
Nateon	Instant messaging	0.9	337.1	62.5
Outlook	MS mail service	12.7	1,150.4	692.3
PuTTY	Telnet/SSH client	0.7	80.4	11.5
Xshell	Telnet/SSH client	1.2	177.1	22.8
Teamviewer	Remote control	1.8	722.6	291.7
Dropbox	Cloud file sharing	11.1	294.8	158.5
uTorrent	P2P download	1,116.4	62,151.0	49,494.5

We divided the ground truth traffic into two different sets that were obtained on different dates. One traffic set was used only for PSS signature generation. The other traffic set was used only for the traffic classification test. **Table 5** indicates the traffic flows for PSS signature generation, and the number of PSS signatures that are generated for each application.

Table 5. Traffic flows for PSS Signature Generation and the Number of PSS Signatures for Each Application

Traffic Class	Flow	# of PSS Signature
Skype	1,000	16
GomTV	2,229	9
Naverlive	1,138	9
Nateon	456	5
Outlook	5,774	49
PuTTY	382	1
Xshell	730	4
Teamviewer	842	9
Dropbox	6,232	4
uTorrent	64,773	192

4.2 Evaluation Metrics

We use completeness and accuracy as evaluation metrics, in accordance with most of traditional literatures [4][5][18]. Completeness is a metric of how much traffic was classified.

Accuracy is a metric that indicates rate of the rightly classified traffic. It is determined by comparing the classification results with the ground-truth. The accuracy is divided into the overall accuracy and accuracy per application which presents the precision, recall and F-measure for each application. These evaluation metrics are expressed by flow, packet, and byte to provide more detailed information.

4.3 Results and Analysis

The proposed method has classified a large amount of the ground truth traffic in just a few minutes, which shows the applicability to real-time traffic classification. **Table 6** presents the overall accuracy and completeness of the traffic classification test in flow, packet, and byte. The results show that the proposed method can achieve high accuracy rates of greater than 99.97% in all units for application traffic that was classified.

Table 6. Overall Accuracy and Completeness

	Overall Accuracy	Completeness
Flow	99.97%	10.03%
Packet	99.99%	62.85%
Byte	99.99%	67.32%

From the completeness perspective, the flow completeness is 10.03%, but the packet and byte completeness are 62.85% and 67.32%, respectively. The classification result of uTorrent that takes up the largest ground truth causes a significant impact on it. The uTorrent traffic takes up to 95% of the ground truth flows, but only 5.8% of the uTorrent flows are used to generate PSS signatures. It is difficult to analyze the remaining 94.2% of flows using the PSS signatures from 5.8% of flows. However, the uTorrent classification results achieve up to 48.60% packet and 54.69% byte recall that affect the packet and byte completeness. This explains that PSS signatures of uTorrent correctly classify heavy flows such as the file download with a lot of packets and bytes, which are more crucial for traffic monitoring and network management [19].

Table 7. Precision and Recall of Each Application

Traffic Class	Flow		Packet		Byte	
	Precision	Recall	Precision	Recall	Precision	Recall
Skype	99.99%	51.49%	99.99%	69.84%	99.99%	85.07%
GomTV	99.97%	5.48%	99.99%	83.57%	99.99%	89.92%
Naverlive	99.99%	80.78%	99.99%	90.84%	99.99%	90.94%
Nateon	99.97%	85.31%	99.99%	98.39%	99.99%	96.19%
Outlook	99.98%	70.85%	99.96%	64.55%	99.96%	57.82%
Xshell	100%	96.28%	100%	97.29%	100%	97.29%
PuTTY	100%	98.13%	100%	99.50%	100%	99.30%
Teamviewer	99.97%	50.41%	99.99%	83.68%	99.99%	83.79%
Dropbox	99.98%	97.01%	99.98%	97.89%	99.98%	97.34%
uTorrent	99.99%	9.12%	99.99%	48.60%	99.99%	54.69%

Table 7 presents the precision and recall per each application, which are responsible for the overall accuracy and completeness. The results show that the proposed method can achieve high precision rates of more than 99.96% for every application in all units, even if a few of recall rates are relatively low that affects completeness.

Each precision rate for every application almost achieves 100%. The flow precision rates are greater than 99.97% for all applications in all units, and packet and byte precision rates are over 99.96% for all of them. Furthermore, all precision rates for PuTTY and Xshell achieve 100%. The misclassification occurred by abnormal behaviors of each application such as packet retransmission and out-of-order. Robust traffic classification to abnormal behaviors of applications is an important topic for our future research.

The low flow completeness problem exists on the flow recall rates of a few of applications such as Skype, GomTV, Teamviewer and uTorrent. The flow recall rate is also less than the packet and byte recall rates for most applications. This is because of flows that have only one or two packets. We defined the minimum value for the dimension of the PSS vector as 3 and generated PSS signatures. However, we used all real traffic traces of the concerned applications in this test and the PSS signatures cannot classify flows with only one or two packets. In addition, flows of TCP sessions that abnormally terminated in traffic traces, affect the low recall rates. GomTV has the largest gap between flow and packet (or byte) recall. This means that the recall of the small sized flows that have a few of packets (or bytes) and are unrelated to the streaming is low, but the recall of the big sized flows that have a lot of packets (or bytes) and are related to the streaming is high.

The recall rate of each application is also affected by the policy for PSS signature conflict. Our method classifies a new flow that is matched to several PSS signatures with different IDs into *unknown* in order to provide high accuracy rates in this study. If other policies are applied, the recall rate can be improved. For example, our method can classify *unknown* into the application of the PSS signature that has the highest similarity [20], additionally use consecutive port numbers [21] or server specific port numbers [22], or report applications of PSS signatures that are matched to *unknown* for the network manager to determine its application. In the future, we will further study the PSS signature conflict to improve the recall rate of each application.

Table 8. F-measure of Each Application

Traffic Class	Flow	Packet	Byte
Skype	67.98%	82.24%	91.93%
GomTV	10.39%	91.05%	94.69%
Naverlive	89.36%	95.20%	95.25%
Nateon	92.06%	99.18%	98.05%
Outlook	82.93%	78.44%	73.26%
Xshell	98.10%	98.63%	98.63%
PuTTY	99.06%	99.75%	99.65%
Teamviewer	67.02%	91.11%	91.18%
Dropbox	98.47%	98.92%	98.64%
uTorrent	16.72%	65.41%	70.71%

Table 8 presents the F-measure for each application. This considers both precision and recall in a single metric using their harmonic mean. The results show that the proposed method

can achieve reasonable F-measures for every application in all units, except GomTV and uTorrent in the flow unit. The low F-measures of GomTV and uTorrent in the flow unit are caused by their low flow recall rates discussed above. However, their F-measures are reasonable in packet and byte units as shown in **Table 8**. This confirms that the proposed method correctly classifies heavy flows with a lot of packets and bytes for GomTV and uTorrent.

4.4 Traffic Classification by Individual Applications

The proposed method can classify traffic into each individual application that is more detailed and useful than the application protocol for network management and operation. Applications that use the same application protocol can be distinguished from each other with PSS signatures, because they communicate with different flow patterns even though they use the same application protocol.

Table 9 shows the packet order and information of the first five packets of the flow that are generated by two applications, Xshell and PuTTY, which use the same SSH protocol. “Server” and “Client” in “Host” field indicate the transmission host that means the direction of the packet. The other fields indicate the payload size in bytes and the description of the packet.

There are distinct differences between Xshell and PuTTY in the direction and payload size of the first five packets based on packet order, although they use the same application protocol (SSH) and data format based on packet order. First, the sizes of the second packet are different from each other. The second packet is sent from client to server, and it contains the SSH version and client program information. The difference in the payload size occurs because the client program information of the two applications is different. The payload size of Xshell is 49 bytes, and that of PuTTY is 28 bytes. Second, the Xshell client transmits only the third packet for “key exchange init”, but the PuTTY client transmits the third and fourth packets for “key exchange init”. Therefore, the payload sizes of the third and fourth packets are different, additionally fourth packet has different transmission direction for the two applications, which affects payload size and transmission direction of successive packets such as the fifth packet.

Due to the differences between Xshell and PuTTY, the traffic of the two applications can be distinguished using PSS signatures. In addition, **Table 9** reminds us that encrypted applications communicate with fixed rules in the first N packets. It means that the proposed method can classify encrypted traffic.

Table 9. Packets of Two Applications, Xshell and PuTTY, using the Same SSH Protocol

Packet order	Xshell Flow			PuTTY Flow		
	Host	Size	Description	Host	Size	Description
1st	Server	21	SSH Protocol info	Server	21	SSH Protocol info
2nd	Client	49	SSH Protocol info	Client	28	SSH Protocol info
3rd	Client	640	Key exchange init	Client	512	Key exchange init
4th	Server	784	Key exchange init	Client	128	Key exchange init
5th	Client	272	Diffie-Hellman key exchange init	Server	784	Key exchange init

Table 10 presents the precision and recall of Xshell and PuTTY from **Table 7** to show that our method can distinguish between two applications that use the same SSH protocol. Xshell

and PuTTY are classified with 100% precision and more than 96.28% recall in all units. No misclassification is why Xshell and PuTTY have clearly distinct flow patterns. In addition, the SSH protocol of Xshell and PuTTY has mostly communicated in the distinct flow patterns that PSS signatures represent, so high recall rates of greater than 96.28% and 98.13% in all units were achieved, respectively.

Table 10. Classification result of Applications, Xshell and PuTTY, using the Same SSH Protocol

Traffic Class	Flow		Packet		Byte	
	Precision	Recall	Precision	Recall	Precision	Recall
Xshell	100%	96.28%	100%	97.29%	100%	97.29%
PuTTY	100%	98.13%	100%	99.50%	100%	99.30%

We conducted a preliminary investigation into whether PSS signatures can be used to identify applications that use the same HTTP protocol through the simple test. **Table 11** shows PSS signatures for three web browsers, Chrome, Firefox, and Internet Explorer, which all use the same HTTP protocol. The PSS signatures are generated from the main flow by accessing to three popular web sites. As shown in **Table 11**, their PSS signatures are different from each other, because their HTTP header fields are different even though they use the same HTTP protocol. In general, the User-Agent and Cookie fields of each web browser are different. Furthermore, there are some cases in which header fields of the reply packet are different, which results in different sizes of the reply packets. Therefore, the web browsers that use the same HTTP protocol can be distinguished by PSS signatures. The accurate and detailed classification method for HTTP traffic will be studied in our future work.

Table 11. PSS Signatures of Three Web Browsers for Three Popular Web Sites

Site URL	Google Chrome	Mozilla Firefox	Internet Explorer
www.google.com	(709, -693)	(546, -496)	(964, -496)
www.yahoo.com	(378, -439)	(338, -439)	(308, -439)
www.facebook.com	(502, -367)	(375, -367)	(326, -402)

5. Conclusion and Future Work

In this paper, we proposed an application-level traffic classification method that uses PSS signature. PSS signature represents the unique flow pattern of each application and can distinguish applications. PSS signatures for each application are generated by our flow grouping, group optimization and signature generation algorithms. After that, our method classifies new flows into individual applications through PSS signature matching in real operation networks.

Our method can be applied effectively to real-time traffic classification in high-speed real operation networks, because it does not require any computation cost for the feature extraction and PSS signature matching is simple similarity comparison. The evaluation shows that our method can classify application traffic easily and quickly with high accuracy rates of more than 99.96% for every application in all units (flow, packet, and byte). The evaluation also shows that our method can classify traffic into each application that uses the same application protocol or encrypts its payload. Therefore, our method can be applied to the various network

management and operation that has to control individual applications with high accuracy in high-speed real operation networks.

Our future studies will focus on three areas. First, we will conduct a study on the PSS signature conflict, to improve the completeness and recall of our method. Second, we intend to extend our method to be robust to abnormal behaviors of applications such as packet retransmission and out-of-order to increase its accuracy. Third, we will study the accurate and detailed classification method for HTTP traffic by extending our method.

References

[1] M. S. Kim, Y. J. Won, and J. W. K. 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)

[2] M. J. Choi, J. S. Park, and M. S. Kim, "An Integrated Method for Application-level Internet Traffic Classification," *KSII Transactions on Internet and Information Systems*, vol. 8, no. 3, pp. 838-856, March, 2014. Article (CrossRef Link)

[3] T. T. T. Nguyen, G. Armitage, P. Branch, and S. Zander, "Timely and Continuous Machine-Learning-Based Classification for Interactive IP Traffic," *IEEE/ACM Transactions on Networking*, vol. 20, no. 6, pp. 1880-1894, December, 2012. Article (CrossRef Link)

[4] A. Callado, C. Kamienski, G. Szabo, B. Gero, J. Kelner, S. Fernandes, et al., "A Survey on Internet Traffic Identification," *IEEE Communications Surveys and Tutorials*, vol. 11, no. 3, pp. 37-52, Third Quarter, 2009. Article (CrossRef Link)

[5] T. T. T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 56-76, Fourth Quarter, 2008. Article (CrossRef Link)

[6] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and Future Directions in Traffic Classification," *IEEE Network*, vol. 26, no. 1, pp. 35-40, January-February, 2012. Article (CrossRef Link)

[7] L. Bernaille, R. Teixeira, and K. Salamatian, "Early application identification," in *Proc. of ACM CoNEXT conference*, 2006. Article (CrossRef Link)

[8] T. Bujlow, T. Riaz, and J. M. Pedersen, "A method for classification of network traffic based on C5.0 Machine Learning Algorithm," in *Proc. of International Conference on Computing, Networking and Communications*, pp. 237-241, January 30–February 2, 2012. Article (CrossRef Link)

[9] 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)

[10] J. Tan, X. Chen, and M. Du, "An Internet Traffic Identification Approach Based on GA and PSO-SVM," *Journal of Computers*, vol. 7, no. 1, pp. 19-29, January, 2012. Article (CrossRef Link)

[11] R. Yuan, Z. Li, X. Guan, and L. Xu, "An SVM-based machine learning method for accurate internet traffic classification," *Information Systems Frontiers*, vol. 12, no. 2, pp. 149-156, April, 2010. Article (CrossRef Link)

[12] S. Runyuan, Y. Bo, P. Lizhi, C. Yuehui, Z. Lei, and J. Shan, "Traffic classification using probabilistic neural networks," in *Proc. of International Conference on Natural Computation*, pp. 1914-1919, August 10-12, 2010. Article (CrossRef Link)

[13] C. Yin, S. Li, and Q. Li, "Network traffic classification via HMM under the guidance of syntactic structure," *Computer Networks*, vol. 56, no. 6, pp. 1814-1825, April, 2012. Article (CrossRef Link)

[14] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," *ACM SIGCOMM Computer Communication Review*, vol. 36, no 2, pp. 23-26, April, 2006. Article (CrossRef Link)

[15] B. C. Park, Y. J. Won, M. S. Kim, and J. W. Hong, "Towards automated application signature generation for traffic identification," in *Proc. of IEEE Network Operations and Management Symposium*, pp. 160-167, April 7-11, 2008. Article (CrossRef Link)

[16] J. H. Yu, H. S. Lee, Y. H. Im, M. S. Kim, and D. H. Park, "Real-time Classification of Internet Application Traffic using a Hierarchical Multi-class SVM," *KSII Transactions on Internet and Information Systems*, vol. 4, no 5, pp. 859-876, October, 2010. Article (CrossRef Link)
[17] F. Gringoli, L. Salgarelli, M. Dusi, N. Cascarano, F. Risso, and K. C. Claffy, "GT: picking up the truth from the ground for internet traffic," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 5, pp. 12-18, October, 2009. Article (CrossRef Link)
[18] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: myths, caveats, and the best practices," in *Proc. of ACM CoNEXT Conference*, 2008. Article (CrossRef Link)
[19] K. C. Lan and J. Heidemann, "A measurement study of correlations of Internet flow characteristics," *Computer Networks*, vol. 50, no. 1, pp. 46-62, January, 2006. Article (CrossRef Link)
[20] R. T. Gu, H. X. Wang, Y. M. Sun, and Y. F. Ji, "Fast Traffic Classification Using Joint Distribution of Packet Size and Estimated Protocol Processing Time," *IEICE Transactions on Information and Systems*, vol. E93.D, no 11, pp. 2944-2952, November, 2010. Article (CrossRef Link)
[21] C. N. Lu, C. Y. Huang, Y. D. Lin, and Y. C. Lai, "Session level flow classification by packet size distribution and session grouping," *Computer Networks*, vol. 56, no. 1, pp. 260-272, January, 2012. Article (CrossRef Link)
[22] J. S. Park, S. H. Yoon and M. S. Kim, "Performance Improvement of Payload Signature-Based Traffic Classification System Using Application Traffic Temporal Locality," in *Proc. Of Asia-Pacific Network Operations and Management Symposium*, September 25-27, 2013.

Jae-Hyun Ham received his B.S. degree in Computer Science and Engineering from Dongguk University, Korea, in 1999, and his M.S. degree in Computer Science and Engineering from POSTECH, Korea, in 2001. He joined the Agency for Defense Development, Korea, in 2001, where he is working currently as a senior researcher in the Department of the 2nd R&D Institute-1. He is also currently a PhD student of Korea University, Korea. His research interests include tactical network management, and traffic monitoring and analysis.

Hyun-Min An received his B.S. degree in Computer Science from Korea University, Korea, in 2012. He is currently a master's student of Korea University, Korea. His research interests include Internet traffic classification and network management.

Myung-Sup Kim 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.