

RESEARCH ARTICLE

Application traffic classification using payload size sequence signature

Kyu-Seok Shim | Jae-Hyun Ham | Baraka D. Sija | Myung-Sup Kim 

Department of Computer and Information Science, Korea University, Sejong, South Korea

Correspondence

Myung-Sup Kim, Department of Computer and Information Science, Korea University, Sejong, Korea.
Email: tmskim@korea.ac.kr

Summary

Recently, network traffic has become more complex and diverse because of 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 signature. The proposed method generates unique payload size sequence signatures for each application using packet order, direction, and payload size of the first N packets in a flow and uses them to identify application traffic. The evaluation shows that this method can classify application traffic easily and quickly with high accuracy and completeness rates, over 99.93% and 93.45%, respectively. Furthermore, the method can classify each application traffic into its respective individual application. The evaluation shows that the method can classify all applications traffic, known and unknown (new) applications into their respective applications, and it can classify applications traffic that use the same application protocol or are encrypted into each other.

1 | INTRODUCTION

Enterprises and campus networks typically impose a set of policies such as quality of service and service-level agreements for an efficient management and operation of network resources. For the policies to be imposed, 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.¹⁻³ Because of 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 These methods were effective and precise in the early days of the Internet.

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 statistic-based classification 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. However, this approach has some issues to be addressed. First, it has to wait until the end of the flow completes 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. Second, it does not consider the incorrect packet sequence at the traffic collection point due to out-of-order, retransmission, and cross-order packets. The incorrect packet

sequence will change statistical information of the flow, which can impact the quality of the statistic-based classification. Third, it classifies application traffic into application classes relatively using to statistical information.

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

The proposed method has 3 advantages. First, it can classify the application traffic in real time and is well applicable 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 computational cost for the feature extraction from the flow. As a result, our method can operate effectively in real operation networks. Second, it can obtain highly accurate classification results with high completeness, because it uses PSS signatures that reflect unique flow patterns of each application and rectifies the incorrect packet sequence due to out-of-order, retransmission, and cross-order packets. Third, it can classify application traffic into each application, not application protocol or type, because it uses specific PSS signatures to each application and identifies application traffic. Our evaluation shows that it can classify application traffic that includes new application not present in signature generation into each concerned application and unknown, and 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 traffic classification approaches and describes the issues of statistic-based classification methods that are recently proposed. Section 3 introduces our proposed application traffic classification 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

In this section, we describe approaches for application traffic classification that have been proposed so far. And we describe the issues of statistic-based classification that has been recently proposed in literature.

2.1 | Traffic classification approaches

To classify Internet application traffic, various methods have already been proposed. These methods fall in 1 of 3 approaches: port-based, payload-based, and statistic-based classification. In this section, we provide a brief overview of the 3 approaches and show their pros and cons.

The port-based classification is the most traditional traffic classification method. It checks the port number of each packet and identifies the application according to the IANA list of well-known and registered ports.^{7,8} Most often, the classification is based on associating a well-known port number to an application. This method needs access only to the header of the packets to find out their port numbers. The payload-based classification method emerged, after the port-based classification become unreliable. It uses specific strings that represent specific application. This method defines these specific strings as the payload signatures and checks whether the traffic contains the signatures or not.⁹⁻¹³ The statistic-based classification method is proposed to address the limitations of port-based and payload-based classification. This method uses statistical properties of network traffic. Its basic idea is that the statistical properties of traffic are unique for different applications and can be used to distinguish applications from each other. The commonly used statistical features contain flow duration, packet inter-arrival time, packet size, bytes transferred, number of packets, etc.

2.2 | Issues of statistic-based classification

Recently proposed statistic-based classification methods commonly use machine learning (ML) algorithms with features such as port number, flow duration, inter-arrival time, packet size, etc that can be characteristics of the application traffic.^{5,6} They have several advantages discussed in the last section. A problem with them is that they cannot classify the traffic in real time because they require flow completion to extract the flow features that can only be conducted at the end of the flow. Despite of few researches on real-time classification using ML methods, it is still hard to avoid complicated computations, which are not very suitable to deploy ML methods in the high-speed real operation networks.

Generally, statistic-based classification uses the packet sequence captured on traffic collection point. However, packets may appear more than once or may be out of order at traffic collection point as consequence of retransmitted or out-of-order packets, which generates the incorrect packet sequence different from the original packet sequence of application. And the packets transferred by the 2 end hosts can cross in the middle of the transmission path due to cross-order packets, which generates different packet sequences depending on the location of traffic collection point. This incorrect and different packet sequence will change the original statistical information of the flow, which can impact the quality of statistic-based classification.¹⁴⁻¹⁷

Packet retransmission occurs when a packet from sender is lost or damaged mainly because of network congestion. If a packet is lost or damaged at its transmission path in network, the sender retransmits the packet by time out or duplicate acknowledgment (ACK). In the situation, the TCP at the receiver stores and uses only the retransmitted packet for application, but the traffic collection system can capture both the lost or damaged packet and the retransmitted packet according to traffic collection point. So, this packet retransmission makes the packet sequence captured at the traffic collection point different from the packet sequence used by the application.

Out-of-order packets are produced because of the various transmission paths travelled by the packets. When a connection is established between 2 end hosts, packets can be transferred through different paths depending on the network status. If a packet that is sent earlier is transferred through more routers than a packet that is sent later, the later packet may arrive at the receiver before the one sent earlier. In this situation, the packets are said to be out-of-order. The TCP at the receiver reorders the packets in the correct order, but the traffic collection system captures them in an incorrect order. As the result, the packet sequence captured at traffic collection point becomes different from the original packet sequence of application.

Cross-order occurs during full-duplex TCP sessions. In a full-duplex TCP session, one end host does not wait for the other host to finish transferring the data. It begins to transfer its own data while the other host is still transferring. Therefore, the packets transferred by the 2 end hosts cross in the middle of the transmission path, which makes traffic collection system capture packets in a different packet order depending on traffic collection point. These cross-order packets can affect packet sequence and statistical information for statistic-based classification at different collection points.

Consequently, it is necessary to correct the packet sequence at traffic collection point according to the original packet sequence intended by a particular application. The proposed method rectifies the incorrect packet sequence because of the retransmitted or out-of-order packets before PSS signature generation or traffic identification, which prevents abnormal packets from affecting the statistical information for traffic classification. The proposed method also deduces the packet sequence in which the application originally sends and receives packets, not the packet sequence at the traffic collection point, for traffic classification.

Most statistic-based classification methods classify the application traffic of the application protocol or type, instead of the application, because they determine the identity of application out of the concerned applications relatively. Figure 1 shows the meaning of the ML-based traffic classification on a 2-dimensional space. Points and their colors indicate flows and traffic classes, respectively. The lines present classification model built from training dataset.

To classify traffic into their respective applications, not by application protocol or group, the proposed method generates PSS signatures that are specific for each application. Payload size sequence signatures are similar to the signatures of payload-based classification, but they use unique flow patterns of each application to classify traffic into their respective applications.

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. To classify applications 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

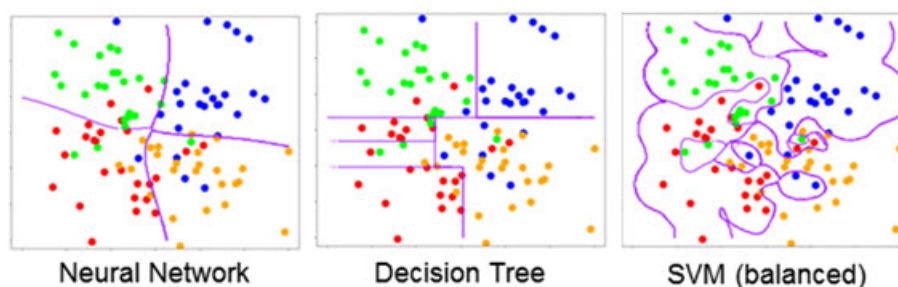


FIGURE 1 Traffic classification on 2-dimensional space^{3,5}. Support vector machine (SVM)

classification method on the basis of the PSS signatures. In this paper, PSS signatures are generated per individual application instead of application protocol or type, therefore can be used in a broad range of fields in comparison with previous methods that classify traffic into application protocol or application type.

3.1 | Payload size sequence

In general, PSS refers to the first few packets of a flow that communicate on the basis of predefined rules by an application. The first N packets of a flow can be used as a distinguishable feature to identify the application since they communicate according to predefined rules and are very different in each application.¹⁸ In this study, we define the flow as a sequence set of packets transmitted in both directions on the basis of a 5-tuple (src IP, dst IP, src port, dst port, and L4 protocol). Packet order is formed according to the packet sequence in which the application originally sends and receives packets at the client host.

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 user datagram protocol (UDP), since 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 “-.” For example, if a flow communicates in both directions as shown in Figure 2 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, and correcting abnormal packets such as the retransmission and out-of-order.

From the observations of popular applications in the Korea University campus network, we found that applications can be distinguished with their PSSs.¹⁹ Figures 3 and 4 show each PSS of 2 different applications, PuTTY, and Xshell. The PSSs that have the same packet order and direction are expressed as a flow group that 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 -1460 to 1460. Most of applications do not have flows that communicate with 1 identical pattern of PSS such as PuTTY. The flows can be either divided with 2 or 3 specific patterns such as Xshell. However, PSSs of each application show their own unique characteristics when they are compared with other applications.

3.2 | PSS signature

A PSS signature represents each application's unique flow pattern that can be used 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 that represents PSS vectors of each group, a distance threshold vector which includes PSS vectors of each group, and a probability distribution set that resolves signature conflict. If all of the individual

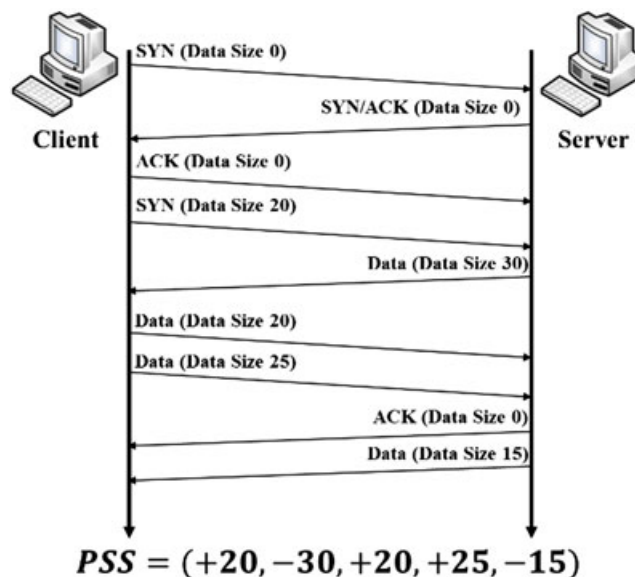


FIGURE 2 Payload size sequence (PSS) of a flow

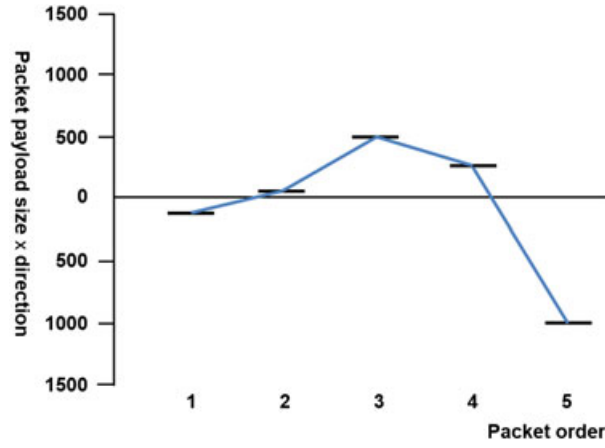


FIGURE 3 Payload size sequences of PuTTY

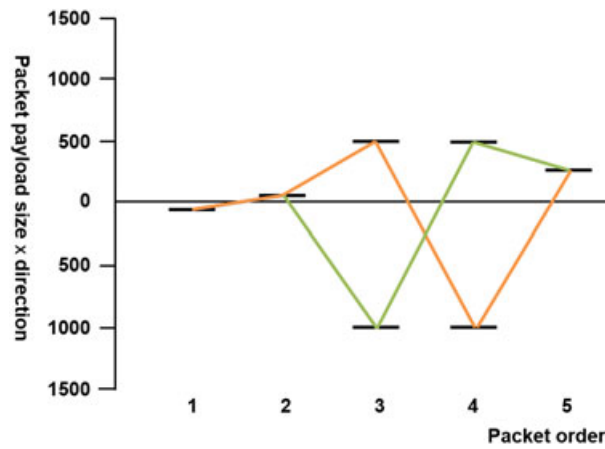


FIGURE 4 Payload size sequences of Xshell

PSS vectors are used as signatures, the number of signatures increases. As a result, managing the signatures becomes difficult; the system as well becomes overloaded for the identification of application traffic. Therefore, an optimal PSS signature combines the representative PSS vector, the distance threshold vector, and the probability distribution set that represent each group by grouping the application PSS vectors.

The PSS vector v_k of a flow f_k can be expressed as Equation 1, and each element $v_{k,i}$ of v_k can be written as Equation 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 . The similarity between 2 PSS vectors is expressed as the distance vector that has elements as distance between 2 PSS vectors by each dimension. Similarity is used in flow grouping and signature-matching algorithms. The distance vector $dv_{x,y} = dv(v_x, v_y)$ between v_x and v_y can be written as Equation 3.

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

$$v_{k,i} = d_{k,i} \times s_{k,i}, \quad (2)$$

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

$$s = (c, t, P), \quad (4)$$

$$dv(v, c) \leq t \text{ for } \forall v \in V(s), \quad (5)$$

$$P = \{P_1, P_2, \dots, P_n\}, \quad (6)$$

$$P(v_k) = \prod_{i=1}^n P_i(v_{k,i}), \quad (7)$$

$$S = \{s_1, s_2, \dots, s_n\}, \quad (8)$$

$$\hat{S} \text{ such that } \begin{cases} \text{minimize } |S| \\ \text{minimize } \sum t(s) \\ \text{maximize } \sum |V(s)| \end{cases} . \quad (9)$$

Payload size sequence signature s is represented with the representative PSS vector, the distance threshold vector, and probability distribution set, as in Equation 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. P is the probability distribution set of the PSS vector group, which can resolve signature conflict. s is the combination of c , t , and P of the PSS vector group.

Each PSS vector v of the PSS vector group $V(s)$, which is represented by the PSS signature $s = (c, t, P)$, should satisfy Equation 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 . The probability distribution set P is a set of the probability distribution P_i that is built on all i th value of every PSS vector belonging to the PSS vector group as Equation 6. If a PSS vector v_k of new flow is matched to several PSS signatures of different applications, which is termed as signature conflict in this paper, the flow is classified according to the probability value calculated from the probability distribution set as Equation 7. Multiple PSS signatures of each application can exist, and Equation 8 presents PSS signature set S . Payload size sequence 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. Applications traffic can be classified according to the PSS signature set.

To obtain an optimal PSS signature set \hat{S} from the given ground-truth flows of each application, the condition of Equation 9 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, Equation 9 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.

Figure 5 presents the flowchart of the proposed application traffic classification method using PSS signature. The method is divided into 2 stages: signature generation and traffic identification. In the signature generation stage, application flows of ground-truth traffic traces are preprocessed to remove control packets, correct abnormal packets, and deduce the original packet sequence. Next, flows of preprocessed ground-truth traffic are vectorized to PSS vectors, and the flows are grouped on the basis of similarity between 2 PSS vectors, which is defined by Equation 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.

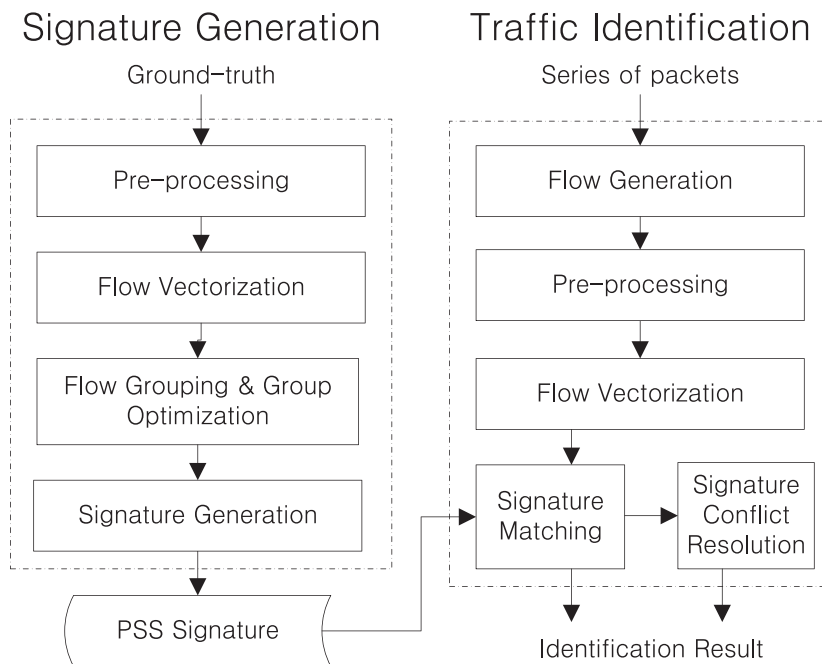


FIGURE 5 Flow chart of application traffic classification using payload size sequence signature

3.3 | PSS signature generation

In this section, we present some requirements of the ground truth traffic to generate optimal payload size sequence signatures. Next, we describe PSS signature generation in detail. Payload size sequence signature generation is composed of preprocessing, flow grouping, group optimization, and signature generation algorithms. The preprocessing algorithm is used to remove control packets, correct abnormal packets, and deduce the original packet sequence. 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 different 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 1 host, the characteristics of traffic can be dependent on that host.

3.3.2 | Preprocessing

Packet retransmission influences the statistical information obtained from a flow because original packets and retransmitted packets can be collected at the traffic collection point, which can change the information, such as the order, size, and number of packets. There are 2 types of packet retransmission, and each has its own elimination process. In the first type, the original packet and the retransmitted packet have the same payload size and Seq number. To eliminate this, we store the original packet and ignore the retransmitted packet to determine the original application behavior. In the second type of packet retransmission, the payload size of the retransmitted packet is larger than that of the original packet because when a packet retransmission is required, TCP repacketizes the packet by adding more data to the original packet to improve the performance.

Algorithm 1. Resolution for Packet Retransmission

```

input P(n) in TCP flow
output TCP flow without retransmission packets
1: P(n): n-th payload packet in a TCP flow
2: P(n).seq: n-th payload packet's sequence number
3: P(n).ack: n-th payload packet's acknowledge number
4: P(n).dir: n-th payload packet's transmission direction
5 P(n).len: n-th payload packet's payload length
6: Find P(k) where P(k).dir == P(n).dir && largest k in 0<k<n
7 if (P(k).seq == P(n).seq )
8 then Delete P(n)
9 else if (P(k).seq + P(k).len < P(n).seq )
10 then Delete P(n)
11 return P(n)

```

Algorithm 1 shows to resolve the packet retransmission. When P(n) is inputted, it is compared with the pre-stored packets that are transmitted in the same direction. If a packet retransmission is detected, P(n) is deleted. The output of algorithm 1 is TCP flow with deleted the retransmission packets. Following the establishment of a TCP connection between 2 end hosts, packets can be transferred through different paths depending on the network status. If a packet that is sent earlier is transferred through more routers than a packet sent that is later, the later packet may arrive at the receiver before the one sent earlier. In this situation, the packets are said to be out of order.

Algorithm 2 shows the algorithm that is used to resolve out-of-order packets. When P(n) is inputted, it is compared with the pre-stored packets transmitted in the same direction. If there is a pre-stored packet with a larger Seq number than P(n), the out-of-order packet is detected and the situation can be resolved. The latest transferred packet that has the same direction and a smaller Seq number than P(n) is P(i). The earliest transferred packet that has the same direction and a larger

Seq number than $P(n)$ is $P(j)$. $P(n)$ is located between $P(i)$ and $P(j)$ when there are no packets in the opposite direction. If packets between $P(i)$ and $P(j)$ are in the opposite direction, the location of $P(n)$ is determined based on these packets. If a packet in the opposite direction is found to have the same ACK number as the sum of the Seq number and the payload size of $P(n)$, then $P(n)$ is placed in front of the packet. If a packet in the opposite direction is found to have a smaller ACK number than the sum of the Seq number and the payload size of $P(n)$, then $P(n)$ is placed behind this packet. The output of algorithm 2 is TCP flow with solved the out-of-order problem.

Algorithm 2. Resolution for Out-of-order Packet

```

input  P(n) in TCP flow
output TCP flow without out-of-order packets
1:  Find P(k) where P(k).dir == P(n).dir && largest k in  $0 < k < n$ 
2:  If P(k).seq > P(n).seq then
3:    Find P(i) where P(i).dir == P(n).dir && P(i).seq < P(n).seq && largest i in  $0 < i < k$ 
5:    Find P(j) where P(j).dir == P(i).dir && smallest j in  $i < j < k$ 
6:    for P(m) from P(j-1) to P(i+1)
7:      if (P(m).dir != P(n).dir && P(m).ack == P(n).seq + P(n).len) then
8:        put P(n) in front of P(m)
9:      Return
11:     else if (P(m).dir != P(n).dir && P(m).ack == P(n).seq + P(n).len) then
12:       put P(n) behind P(m)
13:     return
14:   end if
15:   end for
16:   put P(n) behind P(i)
17: end if
18: return P(n)

```

Packet retransmission and out-of-order packets can be corrected by comparing the Seq and ACK fields of the packets involved. However, there is no correct packet sequence for cross-order. The standard packet sequence is required to remove inconsistency according to traffic collection points and dependency in them. The standard sequence can be the packet sequence at client host or server host. In this paper, we use the packet sequence at the client host as the standard because most TCP sessions start with a request from the client to the server, and they continue with a response from the server to the client. Thus, the session changes continuously according to the requests from the client. Therefore, by inducing the packet sequence in which the client host sends and receives packets, the packets are collected in the original sequence intended by the application.

The Figure 8 occurs when 2 packet crosses are made by 1 backward packet and 2 forward packets. Before the Figure 8 occurs, Figure 6 is always similar to Figure 7. The Figure 8 is detected when 2 or more forward packets and backward packets are crossed. There is Figures 6 and 7 before Figure 8 occurs. Cross-order cases can occur at the same time; therefore, a method that considers all cases is needed to resolve the cross-orders.

Algorithm 3 shows the cross-order resolution algorithm. After detecting cross-order according to its 3 conditions during traffic collection, the algorithm exchanges the order of $P(n-1)$ with that of $P(n)$. At this point, cross-order may also occur between the exchanged $P(n-1)$ and $P(n)$. If cross-orders continue to occur, they are resolved by backtracking until all of the cross-orders are resolved.

Algorithm 3. Resolution for Cross-order

```

input  P(n) in TCP flow
output TCP flow without cross-order packets
1:  If ( P(n-1) == Bp && P(n) == Fp) then
2:    If (P(n-1)(Ack) <= P(n)(Seq) && P(n-1)(Seq) >= P(n)(Ack)) then
3:      Change P(n-1) and P(n)
4:      n = n-1
5:      goto line 1
6:    End if
7:  End if

```

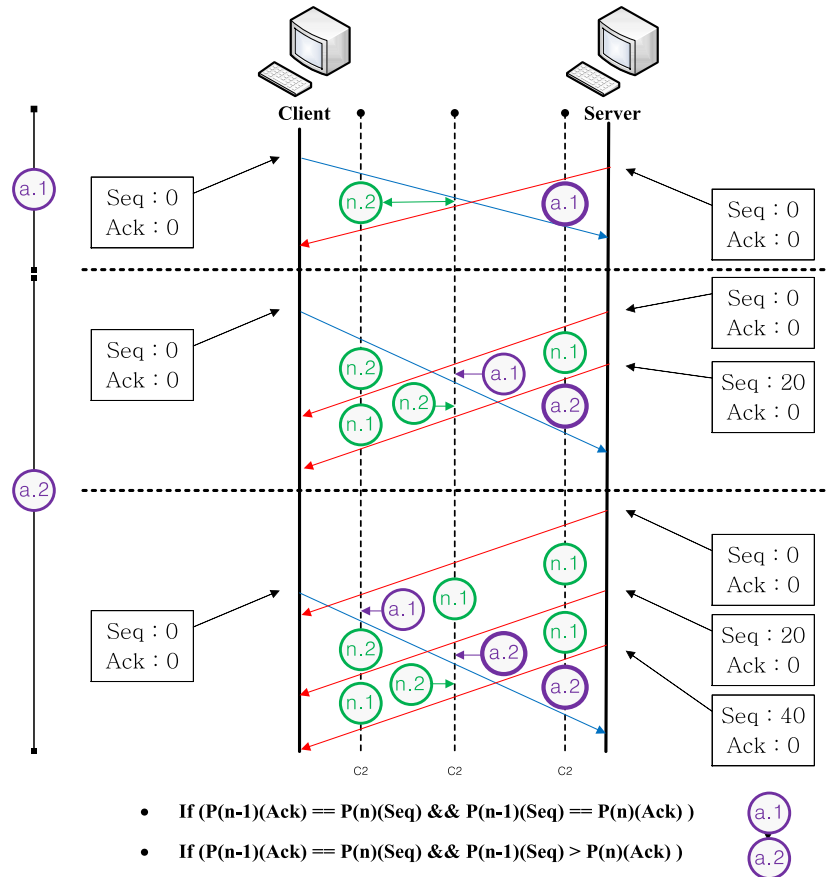



FIGURE 6 Cross-order cases 1 and 2

3.3.3 | Flow grouping and group optimization

First of all, flows from the ground-truth traffic are preprocessed and divided into each application according to the identity of applications. Next, flows of each application are grouped according to similarity of PSS vectors to extract PSS signatures for each application. An application has multiple flow patterns to reflect different application behaviors. To precisely catch these patterns, flow grouping is used for each application. The application flows are divided into multiple flow groups by similarity of PSS vectors to extract PSS signatures of each application from the repeated and unique patterns of flows. Each flow group, as shown in Table 1 has 6 properties.

3.3.4 | Signature generation

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 Equation 12 and assigns it to the C-vector of the PSS signature. The C-vector for PSS signature should be calculated, after the group optimization algorithm removes the PSS vectors that did not meet the conditions given in Equation 10. The C-vector is used with the T-vector to identify the application of a new flow during the traffic classification.

$$c_i = \left(\min(v_{j,1}) + \frac{|\max(v_{j,1}) - \min(v_{j,1})|}{2}, \min(v_{j,2}) + \frac{|\max(v_{j,2}) - \min(v_{j,2})|}{2}, \dots, \min(v_{j,n}) + \frac{|\max(v_{j,n}) - \min(v_{j,n})|}{2} \right) \quad (12)$$

for $\forall v_j \in V_i, c_i$ of G_i

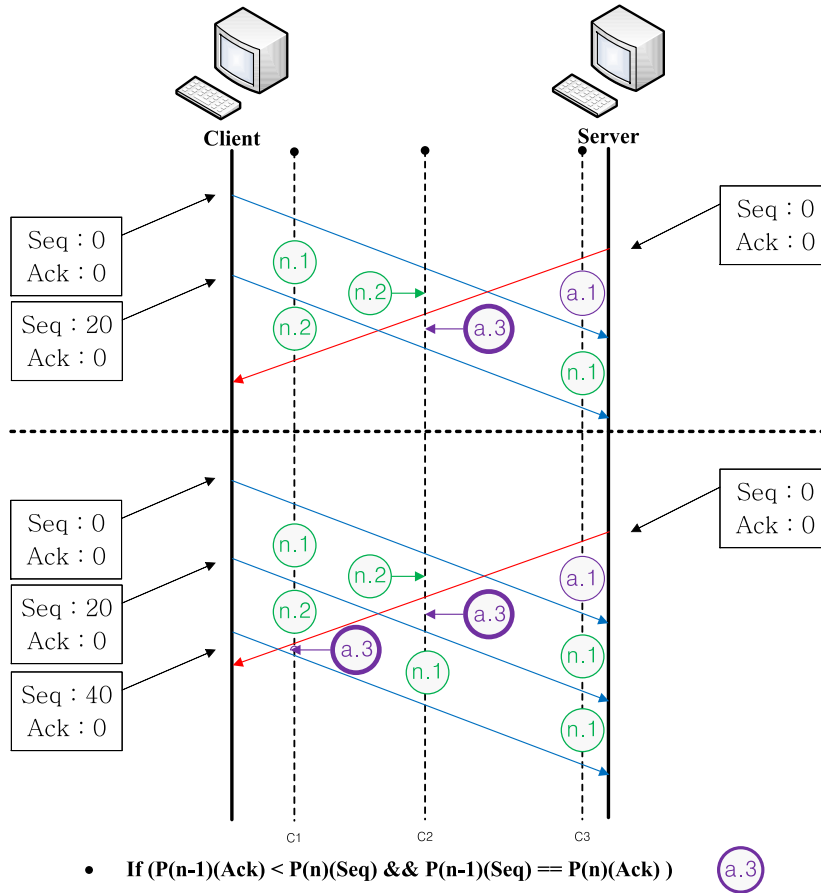


FIGURE 7 Cross-order case 3

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 to eliminate misclassification errors during traffic classification. The T-vector is calculated using Equation 13, 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 its initial value for each group.

$$t_i = (\max(|v_{j,1}-c_{i,1}|), \max(|v_{j,2}-c_{i,2}|), \dots, \max(|v_{j,n}-c_{i,n}|)) \text{ for } \forall v_j \in V_i, c_i \text{ of } G_i \quad (13)$$

Finally, the signature generation algorithm builds the probability distribution set of the flow group and assigns it to the PDS of the PSS signature. The PDS is a set of the probability distribution P_k , which is built on all k th values of every PSS vector in the flow group. The values are already discrete¹⁰ and ranged from -1460 to 1460 bytes, because the k th value of PSS vector is calculated by multiplication of the payload size and direction of k th packet on packet order of each flow. Furthermore, the range of the value in the flow group is much smaller because the value is limited to k th distance value of T-vector. Consequently, we can build P_k as the discrete probability distribution. To build P_k , the signature generation algorithm uses the histogram estimation¹¹ with bin of 1 byte size. It counts the number of the same value of the k th position of every PSS vector in the flow group, and respectively divides them by the number of all PSS vectors belonging to the group. P_k provides the probability for a specific value at the k th position of the PSS vector in the flow group. The P_k of PDS_i of the flow group G_i is built as Equation 14.

$$P_k = \text{Probability distribution of } v_{j,k} \text{ for } \forall v_j \in V_i \text{ of } G_i \quad (14)$$

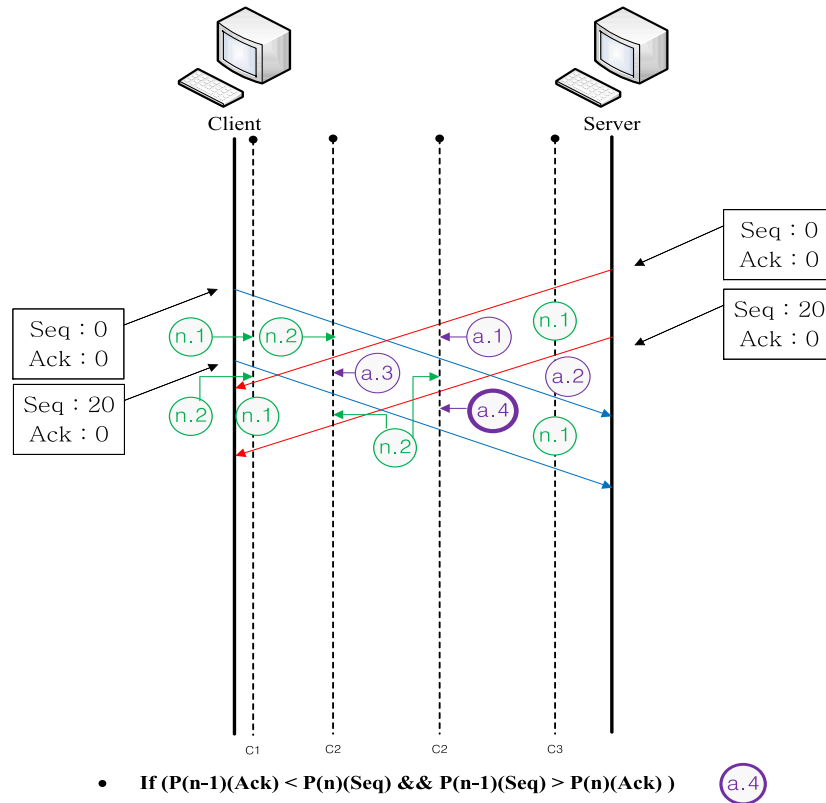


FIGURE 8 Cross-order case 4

TABLE 1 Attributes of a flow group

Attribute	Description	Example
ID	Application name	BitTorrent
Proto	L4 protocol: TCP/UDP	UDP
V	Set of flow vectors	{v1,v2,v3,...,vn}
Dim	Dimension of vector	5
C-vector	Centroid PSS vector	(+20,+30,-50,+20,-30)
T-vector	Distance threshold vector	(175, 175, 175, 175, 175)

Figure 9 shows the PDS of the PSS signature. Payload size sequence is made up of $N P_k$ that is the probability distribution of k th values of PSS vectors in the flow group. P_k is expressed as histogram that is a graphical representation of the distribution of numerical data.

4 | EVALUATION

In this section, we describe the result of traffic classification test in the campus network to verify the proposed application traffic classification using PSS signature. We did only focus evaluation about accuracy and completeness rate of this paper. It will present future work about the result of comparative of classification time and extraction time with conventional method.

4.1 | Ground truth traffic

For our evaluation, we collected bidirectional packet traces from the Korea University campus network. The campus network was configured with 1 router at the Internet junction, so we collected the traffic traces from the router using port mirroring. To

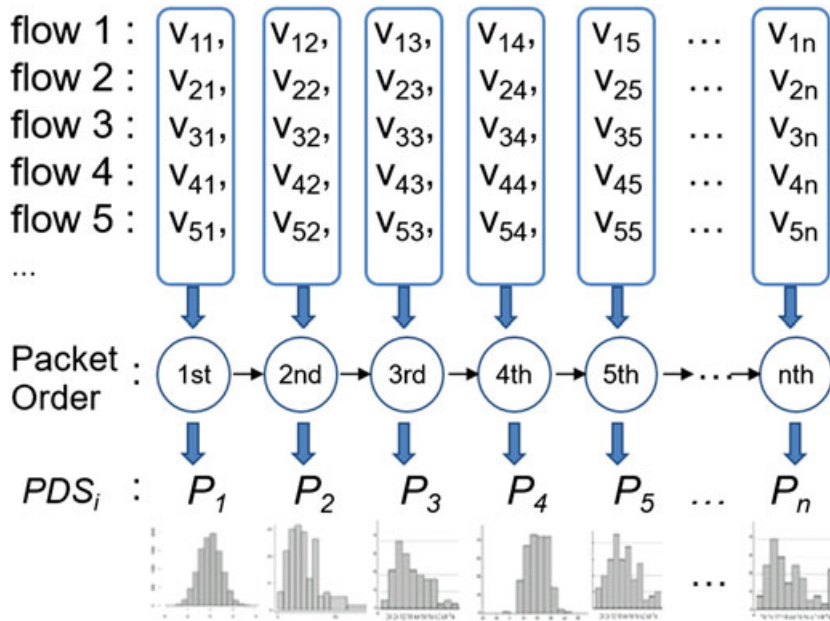


FIGURE 9 Probability distribution set (PDS)

evaluate traffic classification, it is crucial to obtain a firm ground truth. We deployed traffic measurement agents on selected hosts in the campus network and created the ground-truth traffic.²⁰ The ground truth traffic through those agents is more reliable than using the result of the particular classification method to evaluate the classification method.¹²

4.2 | Evaluation metrics

We use completeness and accuracy as evaluation metrics, in accordance with most of traditional literatures.^{4,5,13} Completeness is a metric of how much traffic was classified. Accuracy is a metric that indicates rate of correctly classified traffic. Overall accuracy is the ratio of the sum of all true positives (TP) to the sum of all TP and false positives (FP) for all applications as Equation 18. N is the number of concerned applications. TP is the number of correctly identified objects for a specific application. FP is the number of falsely identified objects for a specific application.

$$\text{Overall Accuracy} = \frac{\sum_{i=1}^N TP}{\sum_{i=1}^N TP + \sum_{i=1}^N FP}, \quad (18)$$

$$\text{Precision} = \frac{TP}{TP + FP}, \quad (19)$$

$$\text{Recall} = \frac{TP}{TP + FN}, \quad (20)$$

$$F\text{measre} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (21)$$

Precision is the ratio of TP over the sum of TP and FP as Equation 19. This metric indicates the fraction of objects that are correctly classified from the total amount of objects classified as a specific application. Recall is the ratio of TP over the sum of TP and false negatives (FN) as Equation 20. This metric indicates the fraction of objects that are correctly classified from the total amount of objects of a specific application. FN is the number of objects that are falsely classified as belonging to another application or unknown for a specific application. $F\text{-measure}$ is the combined form that considers both precision and recall in a single metric by calculating their harmonic mean as Equation 21. 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 few minutes, which shows the applicability to real-time traffic classification. In this section, we analyze the performance of the proposed method and describe the performance of the preprocessing and signature conflict resolution perspectives.

4.3.1 | Performance of proposed method

In this section, we analyze the performance of the preprocessing that rectifies the incorrect packet sequence due to abnormal packets and deduces the original packet sequence, through a comparison without using it. Table 2 presents the overall accuracy and completeness of the traffic classification test in flow, packet, and byte. Unused columns indicate the results of a test without preprocessing. Used columns indicates the results of a test with preprocessing. The results show that the classification using preprocessing can achieve higher overall accuracy and completeness than not using preprocessing. The increase rate in overall accuracy was achieved by correctly identifying the application flows that are falsely identified as different application or unknown by the classification without preprocessing. The increase rate in completeness is achieved by additionally identifying the application flows that are classified as unknown.

Table 3 presents the precision and recall per each application, which is responsible for the overall accuracy. The results show that used can achieve higher precision rates than unused for most applications in all units. Unused falsely classified some of application flows as unknown or different application, because most applications have either retransmission, out-of-order, or cross-order packets. Used correctly classified those flows that were unknown or misclassified in unused using preprocessing. Used achieves an increase of 0.01% to 25.12% in precision rates for applications, in spite of the additionally classified flows as compared with unused. Also, used achieves an increase of 0.03% to 74.4% in recall rates for most applications as compared to unused. In particular, for Xshell, a surprisingly huge increase of 72%, 74.13%, and 74.4% is observed in flow, packet, and byte recall, respectively, as used additionally and rightly classify a large portion of its flows that are unknown in unused. This means that more application traffic can be rightly classified by preprocessing.

TABLE 2 Overall accuracy and completeness of preprocessing perspective

	Overall accuracy (%)		Completeness (%)	
	Unused	Used	Unused	Used
Flow	99.80	99.93	87.42	93.45
Packet	99.89	99.97	94.83	96.19
Byte	99.92	99.99	95.33	96.25

TABLE 3 Precision and recall of each application of pre-processing Perspective

Traffic Class	Precision						Recall					
	Flow (%)		Packet (%)		Byte (%)		Flow (%)		Packet (%)		Byte (%)	
	Unused	Used	Unused	Used	Unused	Used	Unused	Used	Unused	Used	Unused	Used
Skype	99.85	99.85	99.57	99.56	99.8	99.86	80.64	80.64	81.33	81.33	89.03	89.03
GomTV	100	100	100	100	100	100	88.03	96.64	95.20	95.59	95.43	95.68
Naverlive	94.12	97.72	99.97	100	99.97	100	92.75	93.37	97.39	98.74	97.50	98.76
Nateon	73.86	98.98	98.69	99.89	99.11	99.87	98.31	98.31	99.91	99.91	99.64	99.64
Outlook	99.29	99.35	99.86	99.91	99.93	99.96	78.87	88.28	57.14	92.26	52.47	93.57
Xshell	100	100	100	100	100	100	27.14	99.14	24.93	99.06	24.68	99.08
PuTTY	100	100	100	100	100	100	100	100	100	100	100	100
Teamviewer	100	100	100	100	100	100	91.68	91.68	81.21	81.21	81.40	81.40
Dropbox	100	100	100	100	100	100	99.49	99.52	99.21	99.29	99.44	99.56
uTorrent	99.96	99.97	99.84	99.96	99.89	99.99	87.20	93.37	94.49	94.94	95.05	95.09

Table 4 presents the *F-measure* for each application that considers both precision and recall in a single metric using their harmonic mean. The results show that used can achieve higher *F-measure* rates than unused for all applications in all units due to higher precisions and recalls as discussed and experimental results show above. Used achieves an increase of 0.02% to 59.95% in *F-measure* rates for most applications as compared to unused.

4.3.2 | Performance of signature conflict resolution perspective

We analyze the performance of the signature conflict resolutions using PDS through a comparison by using manhattan distance²¹ and classifying them as unknown. Table 5 presents the overall accuracy and completeness of the traffic classification test in flow, packet, and byte. Unk indicates the result of a test that classifies a new flow into unknown in signature conflict. Dist indicates the result of a test using the Manhattan distance between the C-vector and the flow vector for signature conflict resolution. Payload size sequence indicates the result of a test using PDS to resolve signature conflict. The results show that the signature conflict resolution using PDS can achieve high rates in both, overall accuracy, and completeness.

Table 6 presents the precision per each application, which is responsible for the overall accuracy. The results show that PDS can achieve higher precisions than Unk and Dist for all applications in all units. The precision of Unk indicates the precision using only signature matching without signature conflict resolution, because it classifies a new flow that is matched to several PSS signatures with different IDs into unknown. Payload size sequence achieves higher precision rates than Unk for all applications in all units while it additionally classifies application flows in signature conflict as compared with Unk. This means that PDS can correctly classify most of flows in signature conflict contrarily to Dist.

Also, Table 6 presents the recall per each application, which is responsible for completeness. Unk has lower recall rates than Dist and PDS for all applications in all units, since it classifies application flows into unknown in signature conflict. Even though the completeness of Dist and PDS is similar to each other, they have different performance of recall per each application. Dist has 20.39% and 23.97% lower flow recall rates than PDS in Naverlive and uTorrent, respectively. A large portion of Naverlive flows were falsely classified into uTorrent in signature conflict. Some flows of uTorrent were misclassified into Skype, GomTV, Naverlive, or Teamviewer in signature conflict. So Dist do not increase the recall rates of Naverlive and uTorrent in spite of additionally classified flows in signature conflict as compared with Unk. However, PDS increases recall rates for all applications as the amount of flows are additionally classified in signature conflict, as compared to Dist, since

TABLE 4 *F-measure* of each application of preprocessing perspective

Traffic Class	Flow (%)		Packet (%)		Byte (%)	
	Unused	Used	Unused	Used	Unused	Used
Skype	89.22	89.22	89.53	89.53	94.11	94.13
GomTV	93.63	98.29	97.54	97.75	97.66	97.79
Naverlive	93.43	95.50	98.66	99.37	98.72	99.38
Nateon	84.35	98.64	99.30	99.90	99.37	99.75
Outlook	87.91	93.49	72.69	95.93	68.81	96.66
Xshell	42.69	99.57	39.91	99.53	39.59	99.54
PuTTY	100	100	100	100	100	100
Teamviewer	95.66	95.66	89.63	89.63	89.75	89.75
Dropbox	99.74	99.76	99.60	99.64	99.72	99.78
uTorrent	93.15	96.56	97.09	97.39	97.41	97.48

TABLE 5 Overall accuracy and completeness of signature conflict resolution perspective

	Overall accuracy (%)			Completeness (%)		
	UnK	Dist	PDS	UnK	Dist	PDS
Flow	99.90	77.07	99.93	70.96	93.55	93.45
Packet	99.97	91.48	99.97	86.56	96.33	96.19
Byte	99.99	92.45	99.99	87.76	96.39	96.25

TABLE 6 Precision, recall, and *F-measure* of each application of signature conflict resolution perspective

Traffic Class	Precision (%)			Recall (%)			F-measure (%)		
	UnK	Dist	PDS	UnK	Dist	PDS	UnK	Dist	PDS
Skype	99.83	2.42	99.85	69.89	82.88	80.64	82.22	4.70	89.22
GomTV	100	71.92	100	88.26	96.79	96.64	93.76	82.52	98.29
Naverlive	97.05	91.92	97.72	71.64	72.98	93.37	82.43	81.36	95.50
Nateon	98.97	88.18	98.98	96.96	98.31	98.31	97.95	92.97	98.64
Outlook	99.18	99.29	99.35	69.94	88.38	88.28	82.03	93.52	93.49
Xshell	100	100	100	99.14	99.14	99.14	99.57	99.57	99.57
PuTTY	100	100	100	100	100	100	100	100	100
Teamviewer	100	94.58	100	38.95	92.09	91.68	56.06	93.32	95.66
Dropbox	100	100	100	99.52	99.52	99.52	99.76	99.76	99.76
uTorrent	99.96	99.75	99.97	69.35	69.40	93.37	81.89	81.85	96.56

PDS correctly classifies most of them into their applications. Payload size sequence achieves an increase of 0.02% to 68.06% in recall rates for all applications in all units as compared to Unk by correctly classifying most flows in conflict signature. This means that PDS can be more reasonable for signature conflict resolution than Dist.

Finally, Table 6 presents the *F-measure* for each application that considers both precision and recall in a single metric using their harmonic mean. The results show that PDS can perfectly resolve the signature conflict to increase completeness without reduction of accuracy. Payload size sequence achieves an increase of 0.01% to 66.21% in *F-measure* for all applications in all units as compared with UnK. It also achieves an increase of 0.02% to 92.37% in *F-measure* for all applications in all units as compared to Dist, except for Outlook in flow units and Teamviewer in byte units with a slight decrease of 0.03% and 0.16%, respectively.

5 | CONCLUSION AND FUTURE WORK

In this paper, we propose an application-level traffic classification method that uses PSS signature. Payload size sequence signature represents the unique flow pattern of each application and can distinguish applications. Payload size sequence signatures for each application are generated by our preprocessing, flow grouping, group optimization, and signature generation algorithms. After that, our method classifies new flows into individual applications through PSS signature matching and signature conflict resolution 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 computational cost for the feature extraction and uses simple comparison in PSS signature matching and signature conflict resolution. The evaluation shows that our method can classify application traffic easily and quickly with high accuracy and completeness rates of more than 99.93% and 93.45% for every application in all units (flow, packet, and byte). The evaluation also shows that our method can classify application traffic that includes new application not in signature generation into their respective known or unknown applications. The method can classify application traffic that uses the same application protocol or encrypts its payload into another application. Therefore, our method can be applied to various network management environments and operation that have to control individual applications with high accuracy in high-speed real-time operation networks. It also can be easily extended by adding PSS signatures for new applications.

Our future studies will focus on 3 areas. First, we intend to extend our method to the application function level classification to provide more in-depth analysis for traffic monitoring and network management. Second, we will take a deep study on the combination of our method and different methods such as port-based or payload-based classification^{22,23} to improve precision and recall values, for applications with similar application traffic characteristics. Third, we will study the traffic classification method that uses application message size sequence to better reflect application behaviors.

ACKNOWLEDGEMENTS

This research was supported by a Korea University Grant.

REFERENCES

1. Kim M-S, Won YJ, Hong JW-K. Application-level traffic monitoring and an analysis on IP networks. *ETRI J.* 2005;27:22-42.
2. Choi MJ, Park JS, Kim MS. An integrated method for application-level internet traffic classification. *Ksii Trans Int Inform Systems.* 2014;8:838-856.
3. Nguyen TTT, Armitage G, Branch P, Zander S. Timely and continuous machine-learning-based classification for interactive IP traffic. *IEEE/ACM Trans Networking.* 2012;20:1880-1894.
4. Callado A, Kamienski C, Szabo G, Gero BP, Kelner J, Fernandes S, Sadok D. A survey on internet traffic identification. *ommun Surveys Tuts IEEE.* 2009;11:37-52.
5. Nguyen TTT, Armitage G. A survey of techniques for internet traffic classification using machine learning. *Ieee Commun Surveys Tuts.* 2008;10:56-76.
6. Dainotti A, Pescapé A, Claffy KC. Issues and future directions in traffic classification. *IEEE Netw.* 2012;26:35-40.
7. IANA. Port Numbers, <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>, 2017.
8. CAIDA. CoralReef software suite, <http://www.caida.org/tools/measurement/coralreef/>, 2014.
9. Moore A, Papagiannaki K. Toward the Accurate Identification of Network Applications. In: Dovrolis C, ed. *Passive and Active Network Measurement.* vol. 3431. Springer Berlin Heidelberg; Boston, USA, 2005:41-54.
10. Gu RT, Wang HX, Sun YM, Ji YF. Fast traffic classification using joint distribution of packet size and estimated protocol processing time. *IEICE Trans Inf Syst.* 2010;E93D:2944-2952.
11. Pearson K. Contributions to the mathematical theory of evolution. II. Skew variation in homogeneous material. *Philos Trans R Soc Lond A.* 1895;343-414.
12. Gringoli F, Salgarelli L, Dusi M, Cascarano N, Risso F. nd k. c. Claffy, "GT: picking up the truth from the ground for internet traffic," SIGCOMM Comput. *Commun Rev.* 2009;39:12-18.
13. Kim H, Claffy K, Fomenkov M, Barman D, Faloutsos M, Lee K. "Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices," Presented at the PROCEEDINGS of the 2008 ACM CoNEXT Conference. Madrid; Spain; 2008.
14. Bernaille L, Teixeira R, Akodkenou I, Soule A, Salamatián K. Traffic classification on the fly. *SIGCOMM Comput Commun Rev.* 2006;36:23-26.
15. Yan JH, Yun XC, Luo H, Wu ZG, Zhang SZ. Out-of-sequence traffic classification based on improved dynamic time warping. *IEICE Trans Inf Syst.* 2013;E96D:2354-2364.
16. An H-M, Ham J-H, Kim M-S. Method for resolving traffic classification limitations caused by traffic collection points. in the 3rd International Conference on Computational Techniques and Artificial Intelligence (ICCTAI), Singapore, 2014, pp. 22-26.
17. An H-M, Lee S-K, Ham J-H, Kim M-S. Traffic identification based on applications using statistical signature free from abnormal TCP behavior. *J Inf Sci Eng.* 2015;31(5):1669-1692.
18. Bernaille L, Teixeira R, Salamatián K. "Early Application Identification," Presented at the Proceedings of the 2006 ACM CoNEXT Conference. Lisboa, Portugal; 2006.
19. Ham J-H, An H-M, Kim M-S. Application traffic classification using PSS signature. *Ksii Trans Int Inform Systems.* 2014;8:2261-2280.
20. Byung-Chul P, Won YJ, Myung-Sup K, Hong JW. Towards automated application signature generation for traffic identification. in Network Operations and Management Symposium, 2008. NOMS 2008. IEEE, 2008, pp. 160-167.
21. Krause EF. *Taxicab Geometry: An Adventure in non-Euclidean Geometry.* Courier Corporation; 2012.
22. Lin YD, Lu CN, Lai YC, Peng WH, Lin PC. Application classification using packet size distribution and port association. *J Netw Comput Appl.* 2009;32:1023-1030.
23. Park J-S, Yoon S-H, Kim M-S. Performance improvement of payload signature-based traffic classification system using application traffic temporal locality. in Network Operations and Management Symposium (APNOMS), 2013 15th Asia-Pacific, 2013, pp. 1-6.

Kyu-Seok Shim (kusk007@korea.ac.kr) is a PhD degree student in the Department of Computer and Information Science, Korea University, Korea. He received his BS and MS degree in the Department of Computer Science and the Department of Computer and Information Science, Korea University, Korea, in 2014 and 2016, respectively. His research interests include Internet traffic classification and network management.

Jae-Hyun Ham (mjhham@add.re.kr) received his BS degree in Computer Science and Engineering from Dongguk University, Korea, in 1999, his MS degree in Computer Science and Engineering from Postech, Korea, in 2001, and his PhD degree in Computer and Information Science, Korea University, Korea in 2016. 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. His research interests include tactical network management, and traffic monitoring and analysis.

Baraka D. Sija (sijabarakajia25@korea.ac.kr) is an MS degree student in the Department of Computer and Information Science, Korea University, Korea. He received his BS degree in Information and Communication System from Semyung University, Korea, in 2016. His research interests include Internet traffic monitoring and analysis, service and network management, and Internet security.

Myung-Sup Kim (tmskim@korea.ac.kr) is a professor in the Department of Computer and Information Science, Korea University, Korea. He received his BS, MS, and PhD 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 in September 2006. His research interests include Internet traffic monitoring and analysis, service and network management, and Internet security.

How to cite this article: Shim K-S, Ham J-H, Sija BD, Kim M-S. Application traffic classification using payload size sequence signature. *Int J Network Mgmt.* 2017;e1981. <https://doi.org/10.1002/nem.1981>