Game Traffic Classification Using Statistical Characteristics at the Transport Layer

Young-Tae Han and Hong-Shik Park

The pervasive game environments have activated explosive growth of the Internet over recent decades. Thus, understanding Internet traffic characteristics and precise classification have become important issues in network management, resource provisioning, and game application development. Naturally, much attention has been given to analyzing and modeling game traffic. Little research, however, has been undertaken on the classification of game traffic. In this paper, we perform an interpretive traffic analysis of popular game applications at the transport layer and propose a new classification method based on a simple decision tree, called an alternative decision tree (ADT), which utilizes the statistical traffic characteristics of game applications. Experimental results show that ADT precisely classifies game traffic from other application traffic types with limited traffic features and a small number of packets, while maintaining low complexity by utilizing a simple decision tree.

Keywords: Game traffic, traffic classification, traffic measurement.

Manuscript received May 5, 2009; revised Aug. 11, 2009; accepted Sept. 9, 2009.

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2009-(C1090-0902-0036)).

Young-Tae Han (phone: +82 42 350 6250, email: han0tae@gmail.com) and Hong-Shik Park (email: parkhs@ee.kaist.ac.kr) are with the Department of Information and Communications, Korea Advanced Institute of Science and Technology, Daejeon, Rep. of Korea.

doi:10.4218/etrij.10.0109.0236

I. Introduction

The pervasiveness of game environments [1] has activated an explosive growth in game traffic. In [2], it was reported that nearly 4% of all packets in a backbone is from only 6 popular games in the USA, and network computer games based upon P2P models are predicted to make up over 25% of LAN traffic by the year 2010. Game traffic has become one of the most dominant traffic types in current networks. Also, game services have quality of service (QoS) issues [3]. Attention from game service providers has been paid to the game industry market, as this service is one of the most beneficial business areas of the Internet.

Game applications are divided into several genres. The most popular game genres are massive multiplayer online role playing game (MMORPG), first person shooting (FPS), and real-time strategy (RTS) [4]. Their traffic characteristics differ according to their genre. Furthermore, many game services, such as Starcraft and KartRider, have adopted a peer-to-peer (P2P) communication model. P2P models are widely used in many file-sharing applications. These applications have been highlighted during the past several years in the research of traffic classification and modeling due to their high traffic volume, but game services have not. As a result, the traffic characteristics of P2P game applications are little known to the public compared to P2P file-sharing applications. The few known characteristics of games include packet-size distribution and inter-arrival time. Generally, packet size and inter-arrival time are much smaller [5] than in other best-effort services such as HTTP and FTP. Also, games have longer connection times than other services. Figure 1 shows that the average connection time for the top 20 most popular games in Korea [6] is about 35 minutes. However, these characteristics are not

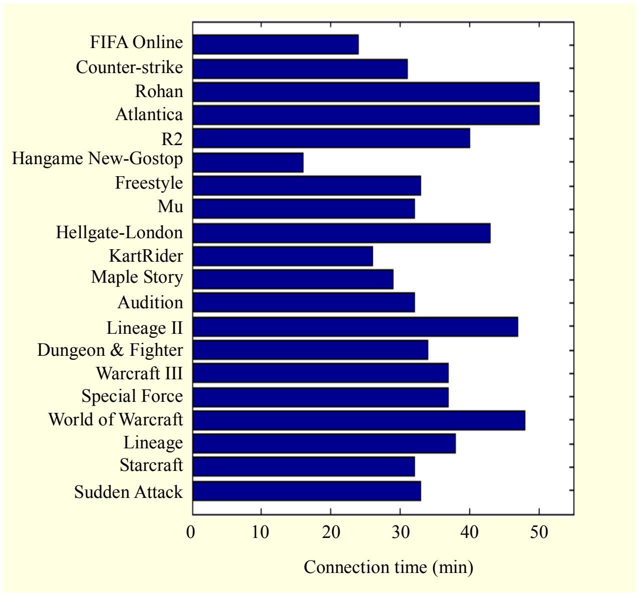

Fig. 1. Average connection time.

enough to classify game traffic efficiently; therefore, it is imperative to further investigate traffic characteristics.

To guarantee QoS in the best-effort network, network management and traffic engineering are essential. For network management and traffic engineering, traffic classification is required [7]. For example, if network providers desire to control traffic from file-sharing P2P, malicious applications such as viruses and malware, and services requiring stringent QoS such as real-time and game services, they should classify traffic. With traffic classification, they can throttle or block high volume traffic applications or malicious applications. Also, traffic classification may be a core element of automated intrusion detection systems [8]. The classification methods include online and offline classification. Online classification determines the application type while a flow is in progress, and offline classification determines the application type when no flow is in progress. There have been many methods for traffic classification of Internet applications. The widely used methods are port-number-based and payload-content-based. Basically, the port-number-based method is very simple, but its accuracy is low, as many emerging applications do not use known fixed port numbers. On the other hand, the payload-content-based method has very good accuracy. The limitations of this method are its complexity and scalability. Furthermore, there is a content protection issue. Currently, some data mining techniques are being used due to the limitations of these two methods [9]-[11]. For data mining or machine learning techniques [12]-[15], their accuracy varies depending on what features are used for classification. Furthermore, many previous studies [9], [12], [13] have mainly focused on P2P applications due to their high traffic volume and traffic dynamic. Little research, however, has been undertaken on game traffic classification in spite of its explosive growth. To classify applications accurately, various types of statistical data are required at the packet or flow level. Moore and others [16] defined many discriminators, but they did not discuss which discriminators are more useful.

In this paper, for the design of a classification method focused on game applications, popular game applications in Korea are examined at the transport layer. Statistically salient characteristics are found, and these are used as key discriminators for traffic classification. One of our findings is that some applications could be simply classified using a set of frequently used packet sizes and its order of elements without complex methods. Also, we observed that distributions of packet sizes for the specific applications are very similar among them, especially in a P2P model; therefore, the correlation values of packet size distributions are utilized in our method. To design a classification method, we exploited offline classification. The reason is that online classification is harder than offline classification because only partial information is available. We propose a traffic classification method for classifying game traffic which utilizes a simple alternative decision tree (ADT) by reflecting statistical characteristics of game applications. ADT has two phases for classifying traffic. In the first phase, statistical data of every flow is examined, and the flow for the P2P model is pre-classified, or the server IP address for the client and server (C&S) model is gathered. In the second phase, flows are grouped using an IP address and port number pair, and these groups are then classified. The benefits of our method are that it does not examine user data so it can be released from forensic issues such as privacy protection, and it precisely classifies game applications with limited traffic features and a small number of packets up to a thousand packets while maintaining low complexity compared to conventional methods.

The remainder of this paper is organized as follows. In section II, related works regarding the characterization of game traffic and application identification or classification are presented. Traffic-traced sets of reference applications and some salient features of game applications are described in section IV. Then, the proposed ADT for classifying game traffic is described in section IV, and experimental results are presented in section V. Finally, section VI concludes the paper.

II. Related Works

In this section, we explain previous works related to game traffic analysis and characterization and present traffic classification methods. There has been little research in game traffic classification; therefore, we begin by describing some common classification methods.

Previously, much attention [18]-[25] has been paid to investigating packet size and inter-arrival time for the analysis and modeling of internet traffic including game applications. In these works, the packet size and packet inter-arrival time are the main features. Many researchers have analyzed Internet traffic from these perspectives and have tried to find suitable statistical traffic models. In game applications, the average packet size is very small, ranging from about 20 bytes to 200 bytes. The known distributions of packet inter-arrival time in [17], [18] are extreme, lognormal, and shifted Weibull distributions. The packet size and packet inter-arrival time are well modeled because traffic patterns of game applications are quite deterministic. The presented games are Quake [17]-[20], Counter-strike [18], [21], and Starcraft [22], [23]. Two of the three, Quake and Counter-strike, are FPS games because FPS games are not only very popular but also very QoS-sensitive applications [24]. The communication models of these games are P2P or C&S. Another popular game genre is MMORPG. In [24] and [25], the characteristics of MMORPGs, such as Shenzhou and Lineage, which are C&S-based applications, are presented. As previously mentioned, many of these works focus on finding proper game traffic models in terms of packet size and packet inter-arrival time and do not aim at classifying game applications.

In studies related to traffic classification, there are two popular methods; port-number-based methods and payload-content-based methods. Traditionally, application identification has been performed using well-known port numbers because most applications use a static port number. Previously, the IANA port number list [26] was used for traffic classification. However, many emerging P2P applications assign a random port number. Furthermore, some applications that use the C&S model also use a random port number assignment. This random assignment makes it difficult to identify applications using a port number. In the case of payload-content-based methods [9], [10], [12], to identify applications, whole or specific parts of the payload content should be examined. This method can identify applications precisely, but it has a scalability problem because it requires large storage and is very complex for finding and mapping a signature. Even though there are sufficient resources and we have overcome these problems, we cannot identify applications if there are no signatures or if the data is encrypted. For game services, most game protocols follow a closed-source, not open-source, proprietary design. Thus, most game protocols are not open to the public. For this reason, it is very difficult to classify game applications, and a new method to classify these applications is needed.

For P2P file sharing applications, many classification methods based on data mining or machine learning techniques are used due to their large traffic volume and traffic dynamics. However, little research [27] tries to classify game applications using machine learning methods. These methods demonstrate good performance in classifying specific applications, but they are still complex to implement. Some other studies [28], [29] use a very simple classification method for limited applications. The authors in [28] identified real-time applications using correlation coefficient values of packet size distributions. This shows that even though packets are encrypted or lost, packet size distributions are highly correlated between those of the same application flows. Although the correlation coefficient value of packet size distributions can be considered one of the key parameters to identify applications, it cannot be a unique parameter. The authors of [29] demonstrate some voice-over-IP (VoIP) application behaviors at the transport layer in terms of packet size.

In summary, most of the previous studies in this area have focused mainly on P2P and best-effort applications rather than game applications. The traffic classification methods are hard to implement due to some limitations including scalability and complexity issues.

III. Statistical Characteristics

To find the appropriate features for traffic classification, we analyzed popular game applications in Korea. In this section, we show the characteristics of game applications at the transport layer in terms of packet-size distributions, bytes per second (BPS), packets per second (PPS), correlations of packet size distributions, and overall statistics of packet sizes.

We chose game applications that are popular and ranked within the top 20 as well as newly emerging game applications, such as Gundam Online (GO) [32], KartRider [33], Starcraft [34], Sudden Attack (SA) [35], World of Warcraft (WoW) [36],

Table 1. Summary of reference game applications.

Application	Type	Architecture	Protocol	Known-port
GO	FPS	P2P full mesh	UDP	3001
KartRider	Racing	P2P full mesh	UDP	N/A
Starcraft	RTS	P2P full mesh	UDP	6112
SA	FPS	P2P hybrid	UDP	27888
D&F	MMORPG	C&S	TCP	10001-10052
MS	MMORPG	C&S	TCP	8585-8587
WoW	MMORPG	C&S	TCP	3724
Lineage II	MMORPG	C&S	TCP	7777

Dungeon and Fighter (D&F) [37], Maple Story (MS) [38], and Lineage II [39] as shown in Table 1. These games belong to very popular game genres such as MMORPG, FPS, RTS, and racing. In addition, these applications use two communication models, namely C&S and P2P. The P2P model can be further divided into two models: full mesh and hybrid. In the full mesh model, one peer communicates with all other peers. When game applications adopt this model, some applications use known port numbers, while some others do not. In our examples, the port number of KartRider is fully unknown because it assigns a port number randomly. In the hybrid model, any peer can be a server, and other peers communicate with it. This is similar to the C&S model, but the server-side IP addresses are very dynamic compared to those in the C&S model. In addition, sometimes the server could be a client in the hybrid model. SA uses a hybrid model and a known port number. For the C&S model, the server addresses are always static. In contrast to P2P-based applications, all of our C&S-based reference applications use known port numbers, but some attractive port numbers like 7777 are shared by some applications. For example, port number 7777 is used by Lineage II, Napster, and Oracle application servers. We collected the reference data at the border router in our campus network with a network processor-based capture system. Our capture system can support links up to 1 Gbps without packet loss. The average utilization of our campus network is about 120 Mbps. To gather reference packets, we first ran these applications at the hosts several times over 30 minutes and collected traffic based on the IP addresses. When we ran game applications, we killed network applications. After that, we eliminated traffic from other applications in reference packets so there would only be traffic from game applications in the reference packets. To avoid an issue regarding private information protection, we did not collect the whole lengths of the packets. We collected 96 byte-length packets including TCP/IP headers and some part of the user's payload data for verification of the classification accuracy.

Figure 2 shows the packet size distributions in the cumulative distribution function (CDF). In presenting these distributions, we divide 1,500 bytes, which is the maximum transfer unit at the IP layer, into 150 bins. In this paper, inbound and outbound are defined as follows. For the C&S model, inbound and outbound can be easily defined using server addresses. For the P2P model, however, it is ambiguous. For the C&S model, inbound is defined as the direction of traffic from the server to the client, and outbound is defined as the direction of traffic from the client to the server. For the P2P model, inbound is defined as the direction of the traffic from other hosts to the host which runs the application, and outbound is defined as the direction of the traffic from one host

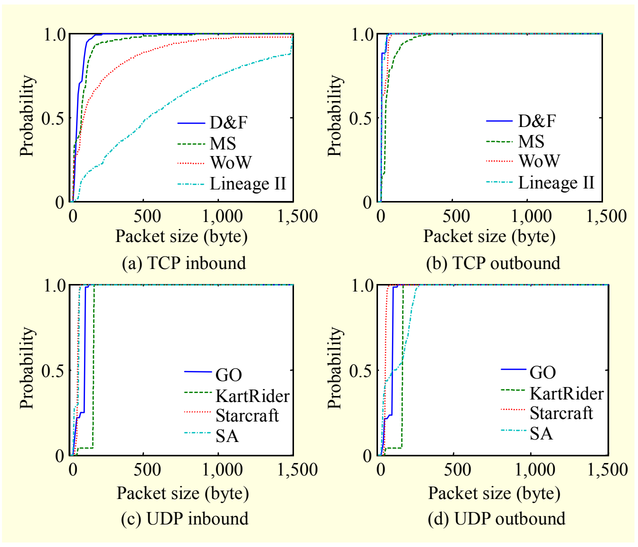

Fig. 2. Packet size distribution (CDF).

to the other hosts. In the C&S model, the average of the inbound packet sizes from a server to a client is usually greater than the average of the outbound packet sizes from a client to a server. Except for the C&S inbound traffic, most of the packet sizes are smaller than 500 bytes, so it is rare to see packets with 1,500 bytes in outbound traffic. KartRider only uses 161 bytes, 56 bytes, and 171 bytes, therefore, its distribution looks like a step function for both directions.

Figure 3 shows the BPS for inbound and outbound traffic. In this figure, we observed that the inbound and outbound traffic volumes are very similar in P2P-based models (GO, KartRider, and Starcraft) except SA. For example, the BPSs of Starcraft are 4.8 and 4.9 for the inbound and outbound direction, respectively. However, in the C&S-based model, the inbound and outbound traffic are asymmetric. For example, the BPSs of WoW are 1,099.3 and 375.3 for the inbound and outbound traffic, respectively. Also, the inbound traffic from a server is always greater than outbound traffic from a client in the C&S model.

Figure 4 shows the PPS for inbound and outbound traffic. The inbound and outbound PPS of UDP- and TCP-based game applications are quite similar. The average PPS is less than 20. For a single connection, the PPS and BPS of UDP-based applications are smaller than those of TCP-based applications. To obtain the total PPS or BPS, we should multiply the number of concurrent players in the case of P2P-based applications.

Thus far, we have presented some traffic characteristics of game applications in terms of packet size, PPS, and BPS. In Figs. 3 and 4, we observed that UDP-based P2P game applications have similar BPS and PPS patterns for inbound and outbound traffic. The difference is quite small. Thus, UDP-

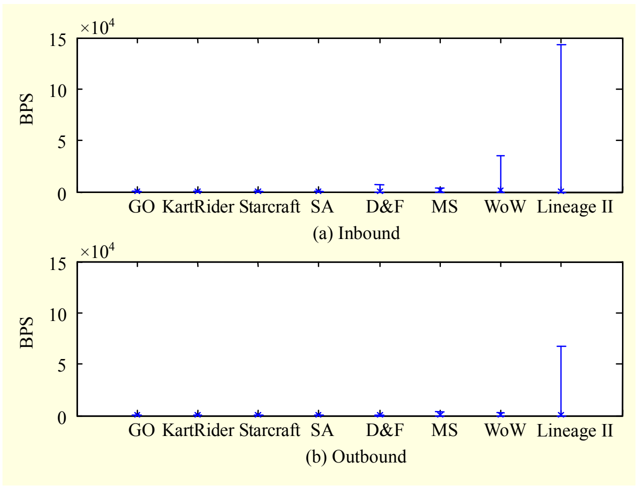

Fig. 3. Inbound/outbound BPS.

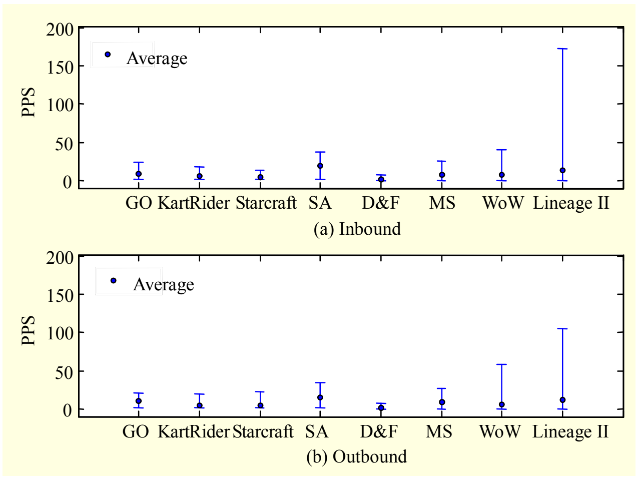

Fig. 4. Inbound/outbound PPS.

based P2P applications have a symmetric traffic pattern in BPS and PPS except for the P2P-hybrid model.

We also examined the correlation of packet size distributions between flows of the same applications. Figure 5 shows the correlation coefficient values. In Fig. 5, correlation values of packet size distributions are calculated between flows from the same application in collected reference data sets. The correlation coefficient is defined as

$$\rho_{nm} = \frac{E\left[(S_n - \mu_n)(S_m - \mu_m)\right]}{\sqrt{Var(S_n)\cdot Var(S_m)}}, \qquad (1)$$

where S_n and S_m are sets of random variables for flows n and m from the same application, and μ_n and μ_m are mean values of S_n and S_m, respectively. We observed that P2P-based applications have very high correlation coefficient values of up to 1. This means that the packet distributions of the same application are almost identical for any flow. However, C&S-based applications have lower correlation coefficient values

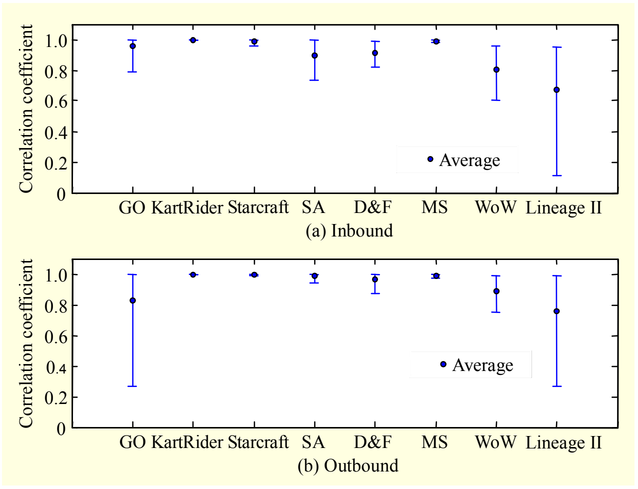

Fig. 5. Correlation of packet size distributions.

compared to P2P applications.

We also examined the cross correlation between different applications. The packet size distributions of some applications are very similar and have high correlation values between these applications. Because we cannot classify some applications due to this high cross correlation, we classify these applications using other statistical characteristics such as min, max, and mode of the packet sizes. For example, D&F and SA have very similar distributions but min, max, and mode are quite different. In statistics, the mode is the value that occurs the most frequently in a data set or probability distribution. In this paper, we define *Mode* as a set of mode sizes for a flow that consists of the most frequently used packet size (*Mode*1) to the third-most frequently used packet size (*Mode*3) as (2):

$$Mode = \{Mode1, Mode2, Mode3\}. \qquad (2)$$

The mode converges when we get more than 250 packets. In our experiments, we used more than 250 packets for each flow. The consideration of frequencies makes classification more complex so we do not consider mode frequencies.

Figure 6 shows the overall statistics in terms of packet size in min, max, average, and mode. We plotted the statistics of 5 flows for each application from the most frequently used packet size (*Mode*1) to the third-most frequently used packet size (*Mode*3). For TCP-based applications, we did not include packet lengths of 40 bytes because this size is used for acknowledgement in TCP. If we include this size, the first ranked mode size is 40 bytes for most TCP-based applications. As seen in these figures, UDP-based applications have almost identical values in min, max, and mode sizes. In particular, the KartRider mode order is the same for every flow. For example, the first ranked mode size is 167 bytes, the second is 56 bytes, and the third is 171 bytes. In some cases, the third ranked mode size is not shown for some flows, but the average packet size is

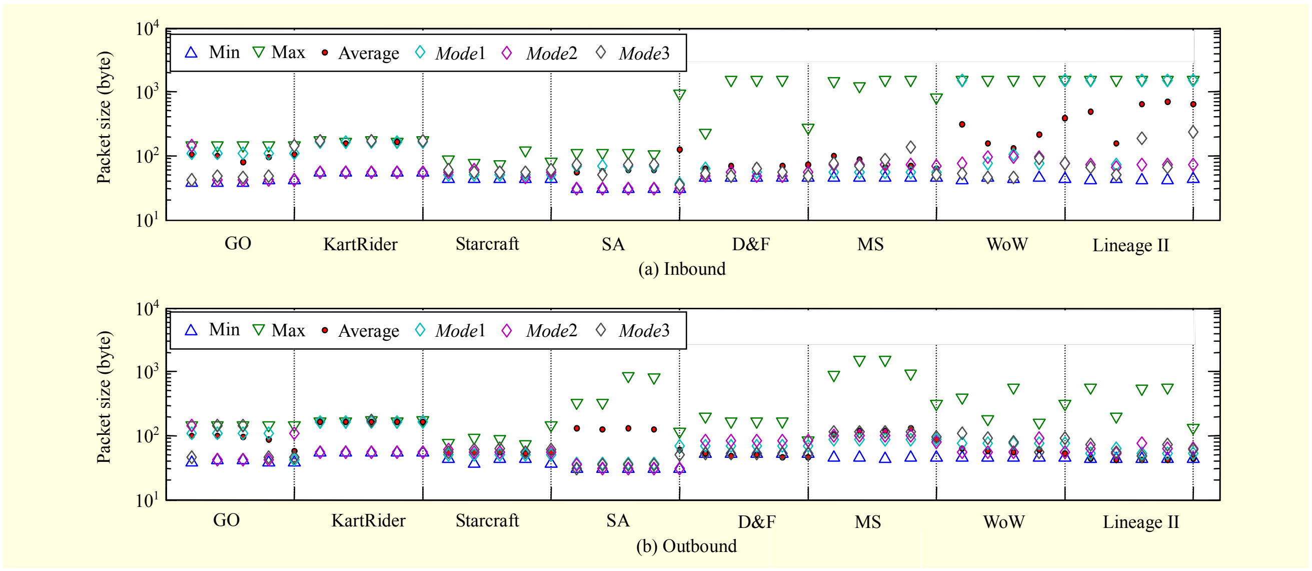

Fig. 6. Overall statistics of packet size.

almost 163 bytes for every KartRider flow. This means that some applications like KartRider can be easily classified using the min, max, mode, and average of the packet sizes. The variation of these statistics is very small in UDP-based applications. The max size of TCP-based applications is quite dynamic for outbound traffic, but this value is around 1,500 bytes for most outbound flows. The behavior of SA is similar to that of TCP-based applications in this respect because it uses a hybrid communication model.

We considered the characteristics of game applications in terms of packet size, BPS, PPS, and correlation. Note that C&S-based game applications have more dynamic characteristics than P2P-based game applications; however, the min, max, mode, and average of the packet sizes and the correlation of their distributions have quite deterministic characteristics. Specifically, for the P2P-based game applications, the flows from the same application have packet size distributions that are very similar to the very high correlation coefficient value. Consequently, these features could be considered critical factors in classifying game applications.

IV. ADT Classification Algorithm

In this section, we explain the proposed classification method, which is based on a simple decision tree with few discriminators. Prior to explaining the proposed classification method, we choose critical discriminators to classify traffic based on our analysis. Table 2 shows several of the critical discriminators for our method. For example, KartRider can be identified by using mode order, and most C&S-based applications can be identified by using only the server-side IP address.

Table 2. Discriminators for classification.

Features	Description
Min, max, mode, mean, distribution	Packet size statistics
Mean BPS, mean PPS	Transport behaviors
Source/destination address, port number, protocol number	Flow information

To compare distributions, we make reference distributions with packet size statistics of reference applications. Each distribution of the applications' packet sizes is obtained using 150 bins, where the bin size is 10 bytes. To find the best distribution fit, the widely used methods are goodness-of-fit tests such as the chi-square and Kolmogorov-Smirmov methods. These tests are appropriate for unbiased distributions such as Gaussian distribution. These tests are not valid to detect or identify biased distribution [40]. The distributions of game applications are quite biased. For instance, several packet sizes are used in game applications especially in P2P-based applications as shown Fig. 2. Thus, these two methods are not appropriate for detecting the distribution. When we use these goodness-of-fit methods with biased distributions, many errors occur. Therefore, the proposed method is designed based upon a correlation-based method rather than these goodness-of-fit methods. Our results show that packet size distributions are highly correlated between flows of the same application. To identify game traffic, we calculate the cross correlation coefficient value between distributions of random flows and references. The cross correlation coefficient between the sampled and reference distribution of packet sizes is defined as ρ_{ij}, given by

$$\rho_{ij} = \frac{E\left[\left(S_i - \mu_i\right)\left(R_j - \mu_j\right)\right]}{\sqrt{Var\left(S_i\right) \cdot Var\left(R_j\right)}}, \tag{3}$$

where R_j is a set of random variables of the reference distribution for application j, S_i is a set of random variables of the sampled distribution for flow i, and μ_i and μ_j are the mean values of R_j and S_i, respectively.

Among the selected features, some have fixed values, such as min and max, but some others, such as mean, vary in specific ranges. For the convenience of handling these values, we define D_{score} as

$$D_{\text{score}} = 1 - \left|\left(\frac{\mu_i - \mu_j}{\mu_j}\right)\right|, \tag{4}$$

which is the normalized distance between the means of R_j and S_i.

Using these features, we make a supervised decision tree called the ADT because nodes can be deleted depending on the applications. Like other decision-tree-based algorithms such as C4.5 [41], the order of the tree nodes is optimized using the concept of information theory in ADT. The order of nodes is quite important because it can affect computation time and complexity.

Here, we briefly describe information gain. The information gain is given as follows [42]. Let us have n values for a given feature, and let T be a set of training samples. Let S be a set of samples, let freq(C_i, S) stand for the number of samples in S that belong to class C_i, and let $|S|$ denote the number of samples in set S. The information entropy is given by

$$\text{Info}\left(S\right) = -\sum_{i=1}^{k}\left(\left(\frac{\text{freq}\left(C_i, S\right)}{|S|}\right) \cdot \log_2\left(\frac{\text{freq}\left(C_i, S\right)}{|S|}\right)\right), \tag{5}$$

where k is the number of possible classes, and C_i is the i-th class. After T has been partitioned into T_i in accordance with n outcomes of one attribute test X, the expected information requirement is

$$\text{Info}_X\left(T\right) = -\sum_{i=1}^{n}\left(\left(\frac{|T_i|}{|T|}\right) \cdot \text{Info}\left(T_i\right)\right). \tag{6}$$

Finally, the information gain is given by

$$\text{Gain}\left(X\right) = \text{Info}\left(T\right) - \text{Info}_X\left(T\right). \tag{7}$$

According to this information gain, we arrange the order of the tree nodes. Thus, we can efficiently prune the number of flows for classification. For example, there are 32,582 flows in one of our tests. When we look for the candidate flows of Starcraft using the min values, we find just 2,161 candidate flows. In this set, there are 838 flows of Starcraft. We can get a similar result when we use mode. However, using mode is more complex than using min values because mode has more elements.

```
Phase I: For every flow:
If ($S_i^{Min} \in R_j^{Min}$){
    If ($S_i^{Mode} \in R_j^{Mode}$){
        If ($Order(S_i^{Mode}) == Order(R_j^{Mode})$){
            If ( $S_i^{Max} \in R_j^{Max}$){
                If ($PPS(S_i) < PPS_{\text{Thread}}$){
                    Store application name of $j$ and set $\rho_{ij}$=1 for flow $i$;}}}
        Else{
            If (($\rho(S_i, R_j) > Val_{\text{Thread}}$)||(($2\text{-}Val_{\text{Thread}} \geq D_{\text{score}}$)&&($D_{\text{score}} \geq Val_{\text{Thread}}$))){
                If (($PPS(S_i) < PPS_{\text{Thread}}$ ) && ($C_{type}$ ==$P2P$)){
                    Store application name of $j$ and $\rho$ for flow $i$;}
                If (($PPS(S_i) < PPS_{\text{Thread}}$ ) && ($C_{type}$ ==$C\&S$)){
                    If (Port number of flow $i$ == Port number of App$_j$ ){
                        Store IP address in $IPT_j$ ;}}}}}}

Phase II: For the flow pairs and IP address:
If ($C_{type}$ ==$P2P$){
    If ($F_i \cap FIS_k$ == $\emptyset$){
        Make a new $FIS_{k+1}\{F_i\{Src\}, F_i\{Dst\}\}$ ;
        Make a new $APNT\{application, \rho_{ij}\}$ for $FIS_{k+1}$ ; }
    If ($F_i \cap FIS_k$ != $\emptyset$){
        If ($F_i^{Src} \notin FIS_k$){
            Add $F_i^{Src}$ into $FIS_k$ ;}
        If ($F_i^{Dst} \notin FIS_k$){
            Add $F_i^{Dst}$ into $FIS_k$ ;
            Add $APNT_k\{application, \rho\}$ ;}
        Find max $\{APNT\{\rho\}\}$ and application name ;
        Identify all flow in $FIS_k$ ;}
If ($C_{type}$ ==$C\&S$){
    IP and port number based classification using IP table ;}
```

Fig. 7. ADT classification algorithm.

The general classification algorithm is shown in Fig. 7. We define C_{type} which has two values depending on the communication model, that is, P2P or C&S. The P2P model includes fully meshed and hybrid models. Here, R_j^{Min}, R_j^{Max}, and R_j^{Mode} are sets of the min, max, and mode packet sizes of the j-th reference application, respectively. The number of elements of R_j^{Min} or R_j^{Max} is a maximum of 2 and that of R_j^{Mode} is 5. In addition, we define two threshold values, Val_{Thread} and PPS_{Thread}. Here, Val_{Thread} $(0 \leq Val_{\text{Thread}} \leq 1)$ is the threshold value of ρ and D_{score}, PPS_{Thread} is the threshold value of PPS, and $\rho(S_i, R_j)$ is the value of the correlation coefficient between S_i and R_j.

As previously mentioned, the key features for classifying the applications are dynamic depending on the application. For example, in the C&S model, the key features are the server-side IP address and port number. If we know these, we can classify C&S-based game applications without error. In addition, these server addresses do not change frequently. However, P2P-based applications have different characteristics. For P2P-based applications, the server-side IP addresses are invalid because any host could be a server, and this server is for temporal purposes. Thus, our algorithm uses different procedures to classify P2P- and C&S-based applications.

Our algorithm consists of two phases. In the first phase, we examine every flow and find the names of the candidate applications for P2P-based and C&S-based applications. If an application uses unique sets of min and mode values, we first check the order of mode values and some other statistical values such as PPS and max values. For KartRider, we classify it using min, max, and mode values and the mode order. We consider this a simple case. For a simple case, ADT does not need any further procedure, so we set the ρ value as 1 to avoid other complex procedures. The remaining unclassified flows keep checking other statistical values such as ρ and PPS. As we showed in the previous analysis, the distributions of packet sizes from the same P2P-based application are quite well correlated. ADT classifies most of the flows for fully meshed P2P-based applications and about half of the hybrid P2P-based applications. However, only a few flows are classified using ADT with previous procedures and the port number. Thus, ADT gathers the IP addresses of the game servers for application j in the IP table (IPT_j) for C&S-based applications.

In the second phase, ADT classifies applications by grouping flows based on the correlation values for P2P-based applications and IP addresses for C&S-based applications. For P2P-based game applications, every flow is grouped using the flow information set (*FIS*). An *FIS* consists of flow information, and the flow information (F_i) is a set of F_i^{Src} and F_i^{Dst} for the flow *i*. Here, F_i^{Src} is defined as the pair of the source address and source port number, and F_i^{Dst} is defined as the pair of the destination address and destination port number. If one element of F_i is included in the previous *FIS*, this flow is grouped into the same *FIS*. On the other hand, if both elements of F_i are not included in the previous *FIS*, a new *FIS* is created. In this manner, we make an *FIS* for every flow. In addition, we define an application name table (*APNT*) for the *FIS*. In *APNT*, we store the candidate application name and ρ for the *FIS*. Finally, we find an application name that has a maximum ρ in *APNT*. We classify other flows in the *FIS* as the same application. For C&S-based applications, we simply classify them using the *IPT*. The IP table contains the detected server-side IP addresses and names of the applications.

V. Classification Results

To verify our ADT, we implemented and evaluated our algorithm using the Perl programming language [43]. We collected the test traffic sets shown in Table 3. We gathered 526 million packets, with a total traffic volume of 382 GB. Date and time were randomly selected.

Table 3. Test sets.

Set	Date	Duration (min)	Number of packets	Traffic volume (GB)
Set 1	03-20-2008	22	7.5e7	64
Set 2	03-23-2008	11	3.4e7	26
Set 3	03-24-2008	34	2.43e8	198
Set 4	03-26-2008	28	1.67e8	91
Set 5	02-27-2008	81	3.7e6	1.7
Set 6	03-20-2008	82	3.4e6	1.6

The UDP-based traffic is quite small at our campus, so we gathered UDP packets only in test sets 5 and 6. Due to the storage and file size limitations on a Linux system, these sets were split into dozens of files. Each file contains thousands of flows, and the average capture time for each file is 1.5 minutes. The size of a split file is 500 MB, and there are about 1K flows that last from the start to end times within the spilt file. Classification was performed using the test data sets shown in Table 3. We examined every flow that has at least 250 packets. ADT is based on packet size distributions so we need sufficient packets for the flows. We examined a total of 106k flows in our test sets.

As performance metrics, we used precision and recall. Given the reference and test sets mixed with flows from game and non-game applications, if a flow of a non-game application is classified as a flow of a known game application with ADT, we denote it as a false positive (FP). Similarly, if a flow of a known game application is classified as a flow of a non-game application with ADT, we denote it as a false negative (FN). A true positive (TP) is the total number of application flows that are correctly classified with ADT. Precision and recall are given by

$$precision = \frac{TP}{TP+FP}, \quad recall = \frac{TP}{TP+FN}. \qquad (8)$$

The server-side IP addresses are critical to classifying C&S-based applications, so we located them.

Figure 8 shows the number of game servers that are identified by our ADT in the reference and test sets. We verified these server IP addresses with known IP addresses or collected IP addresses while we connected to each server. Usually, the server IP addresses of a single application use the same subnet mask. We found one subnet for D&F and MS and two subnets for WoW and Lineage II. We classified the C&S-

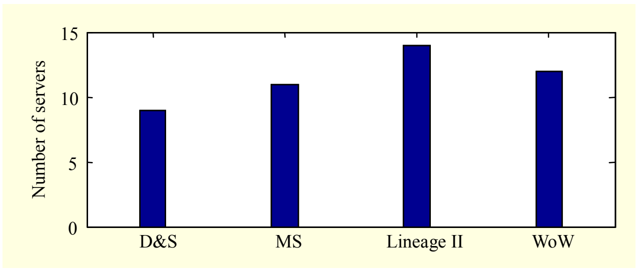

Fig. 8. Number of servers.

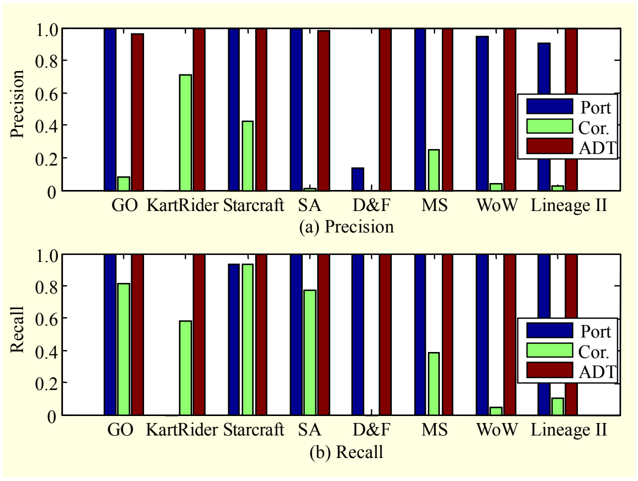

Fig. 9. Overall precision and recall.

based applications with collected server IP addresses by ADT.

Figure 9 shows the classification results in terms of the precision and recall using port-number-, correlation-, and ADT-based classifications. The port-number-based classification uses just the known port numbers. Correlation-based classification uses just the correlation coefficient values. In the correlation-based method, flows are classified if the correlation value between the reference distribution of each application and the distribution of each flow is greater than 0.8, as in ADT. For ADT, Val_{Thread}, D_{score}, and PPS_{Thread} are empirical values. We ran ADT several times to find these values with the best performance. After several experiments, we determine the optimum values of Val_{Thread}, D_{score}, and PPS_{Thread} to be 0.8, 0.8, and 50, respectively. In the case of D&F, it uses a 10001 port number, but there are many other applications that use the same port number and have similar packet size distribution. For this reason, there are many false positives if we do not use IP addresses. In this case, we can classify these flows using the sever-side IP address. C&S-based applications can be classified using the port number and server-side IP address. For a P2P-based game like KartRider, the port number and IP address are not valid for classifying flows of this application. However, it uses several packet sizes. Thus, if we examine mode, min, and max, it can be easily classified without other information. With

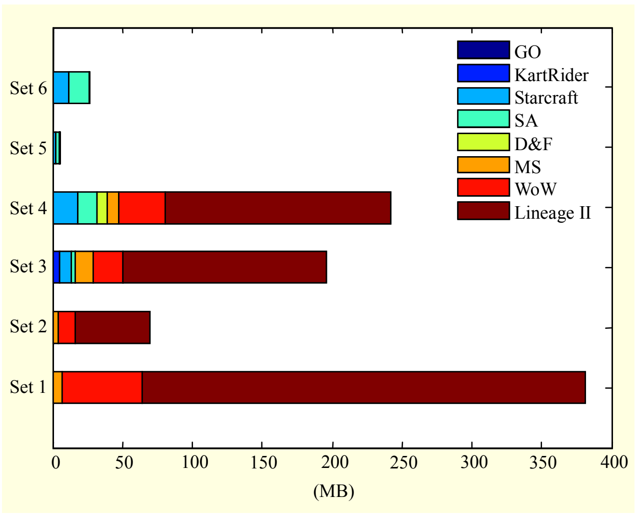

Fig. 10. Traffic volume of each application.

these methods, we examined whole flows in the test sets. As seen in Fig. 9, using only correlation-based methods is not sufficient for classification because there are many similar packet size distributions from different applications. Hence, it has lower accuracy compared to port-number-based methods. A combination of port-number-based and correlation-based methods could produce better results than when we use the port-number-based or correlation-based method alone. However, this combination is not valid for classifying specific applications when their distribution of packet sizes is very similar to that of different applications which use the same port number (that is, WoW, MS, and Lineage II). With ADT, P2P-based or C&S-based applications are accurately classified.

Figure 10 shows the classified game traffic volume in each test set. The largest portion of traffic is for Lineage II. In test sets 5 and 6, there is only UDP traffic, so we can only see Starcraft and SA.

VI. Conclusion

In this paper, in order to classify game traffic, we performed a traffic analysis at the transport layer with several popular game applications including MMORPG, FPS, RTS, and racing games. We found some coincidences between the game applications. One is that the inbound and outbound traffic are quite symmetric with regard to the PPS. In addition, the mean PPS is less than 50, and the average packet size is less than 250 bytes for P2P-based applications and outbound traffic of C&S-based applications. The inbound (server-to-client) traffic of C&S-based applications is quite dynamic compared to outbound (client-to-server) traffic. Furthermore, the packet size statistics show that some statistical features have nearly deterministic characteristics in the min, max, mode, and mean values of the packet sizes for the inbound of C&S-based applications and both directions of P2P-based applications. In this paper, we defined mode as a set of 5 mode sizes. We showed that the distributions of packet sizes and modes of game applications are quite important features for classification, especially in P2P games. Using a few distinctive features, we proposed a highly accurate classification method based on a simple decision tree. As the nodes of a decision tree can be deleted depending on the application's characteristics we call our tree an alternative decision tree (ADT). We showed that ADT can classify each P2P-based application very precisely for every flow without port numbers. However, for C&S-based applications, it has low accuracy when the port numbers are not considered. In this model, traffic from C&S-based applications can be simply identified if the server side IP addresses and port numbers are known. Thus, we used ADT to gather the sever-side IP address for C&S-based applications using the default port number. Using the gathered IP addresses of game servers, ADT precisely classifies each C&S-based application. The benefit of ADT is that it accurately classifies the traffic of game applications using only about 1.5 minute packet traces that contain at least 250 packets. When we examined the flows in our test sets, there were many that had similar behaviors with game applications. If we had more statistical information regarding other game applications, ADT could discover more undetected game traffic and portions of game traffic within Internet traffic. Using ADT, network providers could precisely find traffic volume of games, and they could control game traffic. Furthermore, with accurate classified traffic, network engineers could find more accurate traffic model and workload characteristics. These could be used to design new protocols. Our ADT will be extended toward discovering some other real-time applications, which will be our direction for future studies.

References

[1] K. Jegers and M. Wiberg, "Pervasive Gaming in the Everyday World," *IEEE Pervasive Computing*, vol. 5, 2006, pp.78-85.

[2] S. McCreary and K. Claffy, "Trends in Wide Area IP Traffic Patterns: A View from Ames Internet Exchange," *Proc. 13th ITC Specialist Seminar on Measurement and Modeling of IP Traffic*, Sept. 2000, pp. 1-11.

[3] L. Liang, Z. Sun, and H Cruickshank, "Relative QoS Optimization for Multiparty Online Gaming in DiffServ Networks," *IEEE Commun. Mag.*, vol. 43, May 2005, pp. 75-83.

[4] Game genres. http://en.wikipedia.org/wiki/Game_genres.

[5] G. Armitage, M. Claypool, and P. Branch, *Networking and Online Games: Understanding and Engineering Multiplayer Internet Game,* John Wiley and Sons, 2006, pp. 150-173.

[6] Game chart. http://www.gamechart.co.kr/.

[7] A. Kind et al., "Advanced Network Monitoring Brings Life to the Awareness Plane," *IEEE Commun. Mag.*, vol. 46, Oct. 2008, pp. 140-146.

[8] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," *Computer Networks*, vol. 31, no. 23-24, 1999, pp. 2435-2463.

[9] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures," *Proc. 13th Int. Conf. World Wide Web*, 2004, pp. 512-521.

[10] M.S. Kim, Y.J. Won, and J.W.K. Hong, "Application-Level Traffic Monitoring and an Analysis on IP Networks," *ETRI J.*, vol. 27, no. 1, 2005, pp. 22-42.

[11] J. Erman, M. Arlitt, and A. Mahanti, "Internet Traffic Classification Using Clustering Algorithms," *Proc. ACM SIGCOMM Workshop Mining Network Data*, 2006, pp. 281-286.

[12] T. Karagiannis et al., "Transport Layer Identification of P2P Traffic," *Proc. 4th ACM SIGCOMM Conf. Internet Measurement*, 2004, pp. 121-134.

[13] A. Madhukar and C. Williamson, "A Longitudinal Study of P2P Traffic Classification," *Proc.14th IEEE Int. Symp. Modeling, Analysis, and Simulation*, Sept. 2006, pp. 179-188.

[14] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel Traffic Classification in the Dark," *Proc. ACM SIGCOMM*, Aug. 2005, pp. 229-240.

[15] A.W. Moore and D. Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques," *Proc. SIGMETRICS*, June 2005, pp. 50-60.

[16] A.W. Moore, "Discriminators for Use in Flow-Based Classification," Technical Report, Intel Research, Cambridge, 2005.

[17] M.S. Borella, "Source Models of Network Game Traffic," *Computer Commun.*, vol. 23, no. 4, Feb. 2000, pp. 403-410.

[18] J. Färber, "Network Game Traffic Modelling," *Proc. 1st Workshop Network and System Support for Games*, Bruanschweig, Germany, 2002, pp. 53-57.

[19] T. Henderson and S. Bhatti, "Modeling User Behaviour in Networked Games," *Proc. 9th ACM Int. Conf. Multimedia*, 2001, pp. 212-220.

[20] T. Lang, P. Branch, and G. Armitage, "A Synthetic Traffic Model for Quake3," *Proc. ACM SIGCHI Int. Conf. Advances in Computer Entertainment Technol.*, Singapore, June 2004.

[21] W.C. Feng, F. Chang, and J. Walpole, "A Traffic Characterization of Popular On-line Games," *IEEE/ACM Trans. Networking*, vol. 13, no. 3, 2005, pp. 488-500.

[22] T. Henderson and S. Bhatti, "Networked Games: A QoS-Sensitive Application for QoS-Insensitive Users?" *Proc. ACM SIGCOMM Workshop on Revisiting IP QoS: What Have We Learned, Why Do We Care?* 2003, pp. 141-147.

[23] A. Dainotti, A. Pescapé, and G. Ventre, "A Packet-Level Traffic Model of Starcraft," *Proc. HOT-P2P 2nd Int. Workshop Hot Topics in Peer-to-Peer Systems*, San Diego, USA, 2005, pp. 33-42.

[24] K.T. Chen, P. Huang, and C.L. Lei, "Game Traffic Analysis: An MMORPG Perspective," *Computer Networks*, vol. 50, no. 16, 2006, pp. 3002-3023.

[25] J. Kim et al., "Traffic Characteristics of a Massively Multi-player Online Role Playing Game," *Proc. NetGames*, Oct. 2005, pp. 1-8.

[26] Internet Assigned Numbers Authority (IANA) port number list. http://www.iana.org/assignments/port-numbers.

[27] J. But et al., "Evaluating Machine Learning Methods for Online Game Traffic Identification," *Proc. 5th ACM SIGCOMM Workshop on Network and System Support for Games*, Technical Report 060410C, CAIA, Apr. 2006.

[28] D.J. Parish et al., "Using Packet Size Distributions to Identify Real-Time Networked Applications," *IEE Commun.*, vol. 150, Aug. 2003, pp. 221-227.

[29] K. Tsutomu et al., "Traffic Identification for Dependable VoIP," *NEC Technical J.*, vol. 1, no. 3, 2006, pp. 17-20.

[30] M. Crotti et al., "Traffic Classification through Simple Statistical Fingerprinting," *ACM SIGCOMM Computer Commun. Review*, vol. 37, no. 1, Jan. 2007, pp. 5-16.

[31] N. Williams, S. Zander, and G. Armitage, "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification," *ACM SIGCOMM Computer Commun. Review*, vol. 36, no. 5, Oct. 2006, pp. 5-16.

[32] Gundam-Online (GO). http://gundam.netmarble.net/.

[33] KartRider. http://kart.nexon.com/.

[34] Stracraft. http://www.blizzard.co.kr/starcraft/.

[35] Sudden Attack (SA). http://suddenattack.netmarble.net/.

[36] World of Warcraft (WoW). http://www.worldofwarcraft.com/.

[37] Dungeon and Fighter (D&F). http://df.hangame.com/.

[38] Maple Story (M.S). http://maplestory.nexon.net/.

[39] Lineage II. http://www.lineage2.co.kr/.

[40] L.J. Gleser and D.S. Moore, "The Effect of Dependence on Chi-Squared and Empiric Distribution Tests of Fit," *Annals of Statistics*, vol. 11, no. 4, 1983, pp. 1100-1108.

[41] N. Williams, S. Zander, and G. Armitage, "Evaluating Machine Learning Algorithms for Automated Network Application Identification," *Technical Report 060401B*, CAIA, Swinburne Univ., Apr. 2006.

[42] M. Kantardzic, *Data Mining: Concepts, Methods, and Algorithms*, Wiley-IEEE Press, 2003.

[43] Perl programming language. http://www.perl.org/.

Young-Tae Han received the BS degree from Kyung Hee University, Suwon, South Korea, in 1999, and the MS degree from Information and Communications University (ICU), Daejeon, South Korea, in 2006. He is currently working toward the PhD at the Department of Information and Communications, KAIST, Daejeon, South Korea. His research interests include network performance measurement, multimedia traffic characterization, and traffic anomaly detection.

Hong-Shik Park received the BS degree from Seoul National University, Seoul, South Korea, in 1977, and the MS and PhD degrees from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in electrical engineering in 1986 and 1995, respectively. In 1977, he joined ETRI and worked on the development of the TDX digital switching system family, including TDX-1, TDX-1A, TDX-1B, TDX-10, and ATM switching systems. In 1998, he moved to Information and Communications University, Daejeon, Korea as a member of faculty. Currently, he is a professor of the Department of Electrical and Electronics Engineering, KAIST, Daejeon, South Korea. His research interests are network architecture, network protocols, and performance analysis of telecommunication systems. He is a member of the IEEE, IEEK, and KICS of South Korea.