Performance Improvement of Payload Signature-Based Traffic Classification System Using Application Traffic Temporal Locality

Jun-Sang Park, Sung-Ho Yoon, Myung-Sup Kim
Dept. of Computer and Information Science
Korea University
Korea
{junsang_park, sungho_yoon, tmskim}@korea.ac.kr

Abstract— **Traffic classification is a preliminary and essential step for providing stable network services and efficient network resource management. However, the payload signature-based method has a major drawback when used in high-speed network environment; it has a lower processing speed than other classification methods. This is despite the fact that it has the highest performance in terms of accuracy, classification completeness, and practicality. In this paper, we propose a cache-based traffic classification method that uses application traffic temporal locality and an efficient cache management system to improve the processing speed of the payload signature-based traffic classification. The proposed method achieves an approximately 10-fold increase in processing speed and a 10% improvement in classification completeness over the payload-based classification system.**

Keyword—Traffic Classification; Payload Signature

I. INTRODUCTION

As individual and corporate users are becoming increasingly dependent on the Internet, network speeds are increasing and a variety of services and applications are being developed. Thus, there is a growing need for monitoring and analyzing Internet traffic from an application perspective, in order to achieve efficient network operation and management in various areas such as pay-for billing, CRM (Customer Relationship Management), SLA (Service Level Agreement), etc. Further, the need for traffic monitoring and analysis will continue to increase. Effective methods are needed for analyzing many types of application-level traffic. They are also required for handling real-time processing of large amounts of traffic on high-speed links.

* This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (2010-0020728) and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2007483)

Traffic classification is a preliminary and essential step for efficient network resource management and the provision of stable network services. While a number of classification methods have been introduced in the literature, the payload signature-based classification method shows the highest performance in terms of accuracy, completeness, and practicality [1, 2, 3, 5]. However, the processing speed of the current payload signature-based classification system is insufficient for real-time handling of the huge amounts of traffic data in high-speed networks [7, 8]. Given the increasing number of applications as well as the increasing use of applications that generate large amounts of traffic, the inadequate processing speeds of payload-based analysis is a problem that must be solved.

Many types of application traffic are generated from the target network link, but most of the traffic is generated by a relatively small number of applications. The popularity of various applications could differ significantly. This is because certain websites, email services, etc., are more popular than other services. From a survey of the traffic in a campus network, we found that the average on the whole observation period of the TCP flow to the server end-point ratio is about 20. This means that 5 flows connect to the same server end-point [6]. In this paper, we define this phenomenon as a "temporal locality" of the application traffic. We propose a server IP, Port cache-based traffic classification method to improve the processing speed of the payload signature-based classification systems. In addition, we propose an efficient method to manage the server IP, Port data in the cache.

The remainder of this paper is organized as follows. Section 2 describes related research. Section 3 defines the motivation of this study, namely, the temporal locality of application traffic. In section 4, our solution based on the experimental results is suggested to improve the processing speed. In section 5, the proposed method is applied to our classification system and its validity is proven. Finally, Section 6 describes conclusions and future research directions.

II. Related Work

Many applications try to bypass the firewall for a seamless service by frequently changing their traffic patterns. For this reason, the signatures that identify these applications from traffic data appear in complex forms [4]. In addition, due to the increase in network-based applications and L7 protocols, the number of signatures necessary for identifying applications has been increasing. As the number of signatures and their complexity increase, the processing speed of the payload signature-based classification becomes an important element in determining the performance of the traffic classification system. Many ongoing studies on payload signature-based classification systems aim to accelerate the classification process. However, most studies focus on improving the performance of the pattern-matching algorithm.

Linux L7-filter and Snort are widely used for application-level traffic classification. The L7-filter uses regular expressions to represent signatures and NFA (Non-deterministic Finite Automata) for pattern-matching. However, when over 70 signatures are used to classify applications, the processing speed falls to 3.5 Mbps or less [7]. DFA (Deterministic Finite Automaton) has been proposed to increase the processing speed of NFA. Snort applies the DFA-based pattern matching method, but it has a processing speed of less than 100 Mbps [7], [8]. These two systems tend to focus on pattern matching algorithms to improve performance. However, the time complexity of the matching algorithm is wholly dependent on the configuration of the input data, resulting in limited performance improvement. Thus, real-time traffic analysis of high-speed links (Gbps) might be insufficient if they are only based on the elaboration of the matching algorithm. Thus, we must consider other options to improve the processing speed of the traffic classification system.

Abhishek Mitra et al. [10] implemented an NFA-based regular expression engine on an SGI Altix 4700 workstation with FPGA (Field-Programmable Gate Array) support for hardware solutions. It significantly improves the throughput of NFA, while maintaining a compact memory requirement. However, these approaches require either specialized hardware or computers with high processing power and memory size. Hardware-based solutions usually have higher production costs and are more difficult to upgrade to support new applications. Therefore, this paper focuses on a software solution to improve the processing time of the payload signature-based classification system.

We propose that a history of already classified flows should be kept to build a knowledge base of particular server IP, port combinations. This method can improve the processing speed and the completeness of classification.

III. Application Traffic Temporal Locality

In this section, we define the application traffic temporal locality using occurrence of traffic patterns. Additionally, we prove its validity via experimental evaluation on our campus traffic trace.

A. Traffic Trace

This segment describes the traffic configuration that proves the validity of the proposed method and the temporal locality of the application traffic.

Table 1 shows the result of the traffic trace used in the experiment. Traffic was collected at the Internet connection point of our campus network. It was composed of traffic from a variety of applications that were used by approximately 3000 hosts.

Table 1. Traffic Trace

Measure	Flows	Packets	Bytes
Volume	51,477K	2,012M	1,578GB
Duration	2012.09.12 00:00–23:59		

Table 2 indicates the application use in terms of flow, which is measured using the baseline classification system [1].

We developed this system and deployed it in our campus network for real-time classification of campus Internet traffic. A total of 845 payload signatures were extracted via the payload-signature extraction system. The system specification was Intel ® Core2 Duo E7200 2.53GHz CPU with 4 GB memory. The average processing speed of our system was 160 Mbps of Internet traffic. This speed is insufficient to support the link rate (up to 350 Mbps) of our campus Internet traffic.

Our signature-based classification system achieved more than 99% accuracy and 85% completeness in terms of flow/packet/byte for the entire Internet traffic on the campus network.

The trace contained mainly web services and P2P (Peer-to-Peer) traffic, generated by the torrent “donkey.” Most of the traffic was generated by a few applications in the target network. If we can reduce the processing speed of classification for these applications, the total processing speed of the classification system can be improved.

Table 2. Breakdown of Top 10 Applications

Top 10	App. Type	Flows	Packets	Bytes
1	p2pfilesharing	25,395K	561,709K	356GB
2	web browser	17,043K	1,037,368K	864GB
3	im	1,086K	3,0381K	16GB
4	utility	973K	4,1905K	32GB
5	multimedia	759K	20,5534K	172GB
6	game	691K	31,088K	18GB
7	sns	449K	9820K	6GB
8	security	248K	6,283K	4GB
9	vaccine	230K	28,970K	28GB
10	commercial	126K	22,358K	20GB

B. *Distribution of Server IP, Port*

Figure 1 shows the number of TCP flows and the corresponding number of server IP, Ports for each hour on the link connecting our campus network.

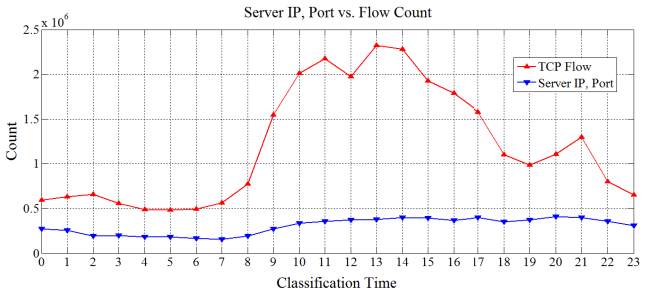

Figure 1. SSIP vs. Flow

The average of the flow to service ratio is about 30% for the campus trace, when the entire observation period is taken into consideration.

A flow transmitting data between hosts has two end-points: the client end-point and the server end-point. Each of these can be represented by the triple of IP address, transport-layer port, and transport-layer protocol. We define the SSIP (Same Server IP Port) as the set of flows with a common server endpoint. The flows belonging to one SSIP are usually generated by the same application.

Figure 2 shows the CDF (Cumulative Distribution Function) graph that represents the flow to SSIP rate of the campus traffic trace.

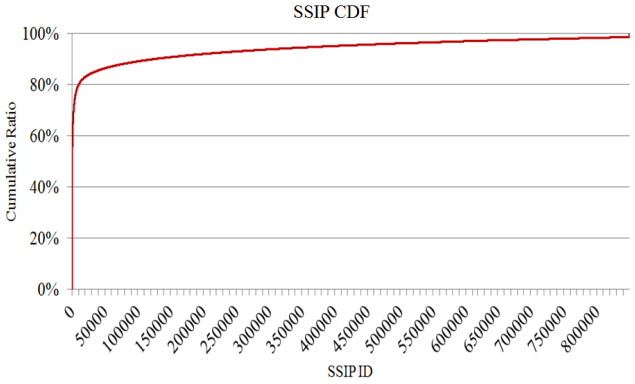

Figure 2. SSIP CDF

The trace associated more than 80% of the flows to only 10,000 SSIPs. Since flows in each SSIP are generated by the same application, a traffic classifier leveraging SSIP is proposed in order to achieve better processing speed and classification completeness than payload signature-based methods.

Application traffic temporal locality means that an SSIP that has just been referenced has a high probability of being referenced again in the near future. Increased application traffic temporal locality generally improves the performance of the SSIP cache-based classifier.

IV. Proposed Method

In this section, we propose our solution to improve the processing speed of the payload signature-based traffic classification system. We aim to prove the validity of these solutions via experiments.

A. *Effect of SSIP Cache-based Classification*

Figure 3 shows a conceptual diagram of the proposed traffic classification methodology.

The server IP and port is listened to by the server-side of an application. A server IP and port is generally multiplexed between a number of different flows, which means only one flow needs to be classified and all the others can share the same classification.

Clients A and B are connected to the same server via the same application port. At this point, if the flow generated by client A is identified through the payload signature-based method, the flow generated by client B is identified using the server IP, port information already obtained, without analyzing the payload signature-based classification. If a flow has been identified by the payload signature-based classification, then the server IP, port address of the flow is updated in the cache table.

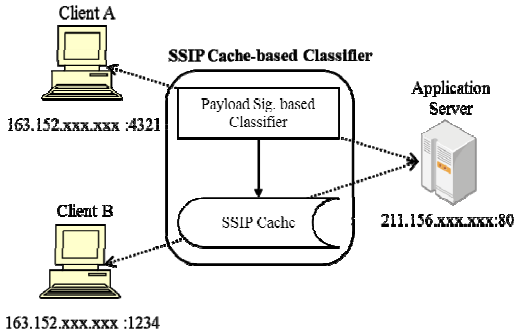

Figure 3. Conceptual Diagram of the Proposed Method

Figure 4 represents the traffic percentage that the payload signature-based traffic classification and SSIP cache-based traffic classification could classify. The area A represents the traffic percentage that the payload signature-based traffic classification could classify, whereas the area B represents the traffic percentage that the SSIP cache-based traffic classification could classify.

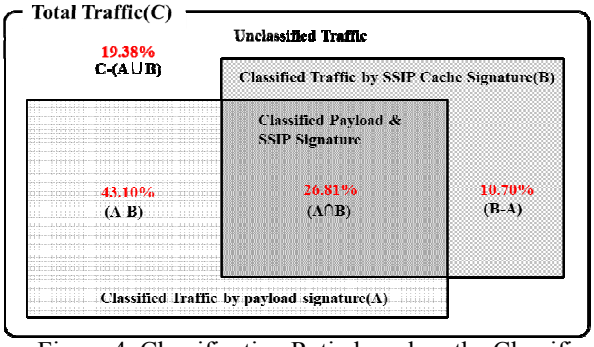

Figure 4. Classification Ratio based on the Classifier

Area A∩B represents the percentage of traffic for which the classification system processing speed can be improved. The payload signature-based traffic classification classifies flows by matching the payload signature of A∩B. If the SSIP cache is used, it can improve classification speed because it classifies by matching only the Server IP, Port. This method could classify 26.81% of the total traffic flow by matching server IP, ports in the campus network.

Area B-A in Figure 4 represents the additional traffic flows that the suggested traffic classification method could classify. This method can improve the completeness of the classification system because it uses the SSIP cache to identify the flows that could not be classified via the payload signature-based classifier.

B. Implementation

The goal of this design is to achieve a much better processing speed and completeness than the payload signature-based classification system by leveraging the SSIP cache.

Figure 5 shows a flow chart of the proposed SSIP cache-based traffic analysis method.

The classification system is mainly composed of two parts: the Signature Match part and the SSIP Cache Manage part.

The Signature Match part is composed of a SSIP cache-based matching module and a payload signature-based matching module.

The SSIP cache-based classifier looks up the cache data to determine the application name. When the flow does not find a match in the SSIP cache, it is sent to the payload signature-based classifier for application identification based on its payload. The SSIP Cache Manage part could improve analysis rate result from minimized analysis time through insert, update, delete, reordering of SSIP.

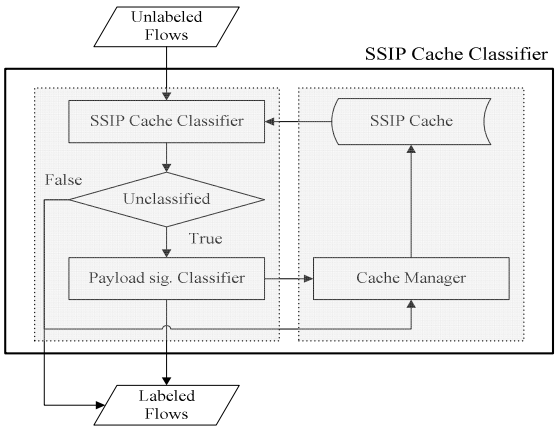

Figure 5. Flow Chart of the SSIP Cache-Based Traffic Classification Method

C. Cache Manager

To design an efficient SSIP cache, we need a memory structure that is optimized for both cache lookup and cache replacement operations. In addition, an efficient cache replacement algorithm is necessary to get a high cache hit ratio using the limited memory space. We demonstrate the memory structure, the cache replacement algorithm, and the cache rearrangement method in the following subsections.

Figure 6 describes the Cache Manager function. The cache manager conducts insert, update, delete and reordering of the SSIP.

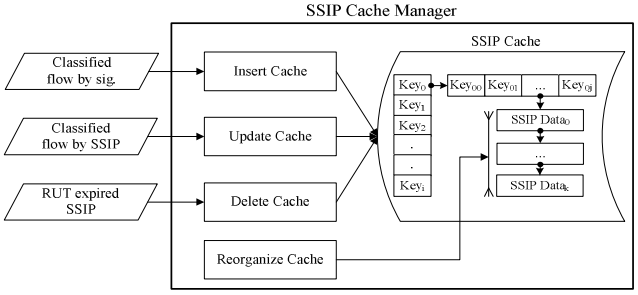

Figure 6. Cache Manager

The "Insert Cache" module of the cache manager is the part that saves the classified SSIP of flows and application names by using the payload signature-based classification. The "Update Cache" module of the cache manager updates the used-time at last analysis and the hit count of classification. This module requires information regarding cache deletes and memory structure rearrangement. The "Delete Cache" date delete module conducts deleting old item. Lastly, the "Reorganize Cache" module performs memory rearrangement in order to improve search speed by searching for frequently used SSIP information first. This helps to offset the decrease in cache search speed when a large number of dates are saved in the cache.

- *Cache Memory Structure*

The SSIP cache store accumulates a large amount of information because the traffic classification system operates continuously in real time. If a large amount of SSIP information is accumulated, it causes a problem where the cache-based traffic classification speed decreases. Therefore, it needs a memory structure for effectively managing the cache memory.

We propose a dual hash memory structure that overcomes the limitations of a single hash memory structure.

Figure 7 shows the dual hash memory structure. α and β are the sizes of the first level hash table and the second level hash table respectively. The value of α is determined according to the amount of data. In a previous study, our research group proposed a method that calculates the size of the first level hash table [11]. The first level hash function uses only 3 bits that combines the last 8 bits for Server IP, Port and L4 protocol. The size of the second level hash table is defined as 65,536 (2^{16}) in light of the campus traffic volume. The second hash function folds the server IP, the sum of server port and the L4 Protocol.

The dual hash memory structure decreases collision of hash keys via a 2-level hash key creation method. It can provide scalability of the classification system by checking the amount of traffic before loading to memory. The dual structure is an efficient cache memory structure for

determining the variable hash table size when the network traffic volume is flexible.

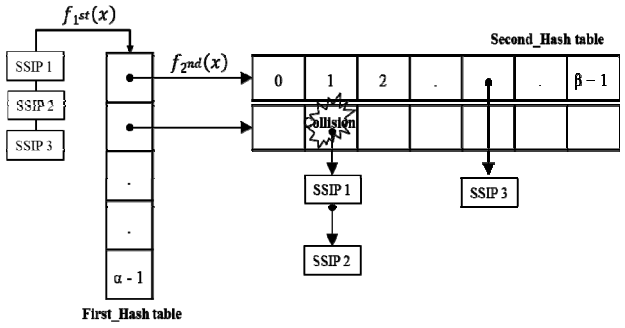

Figure 7. Cache Memory Structure

Figure 8 represents the total comparison frequency when a data search of the total traffic is performed. This is done by applying single hash and dual hash techniques for the same traffic data.

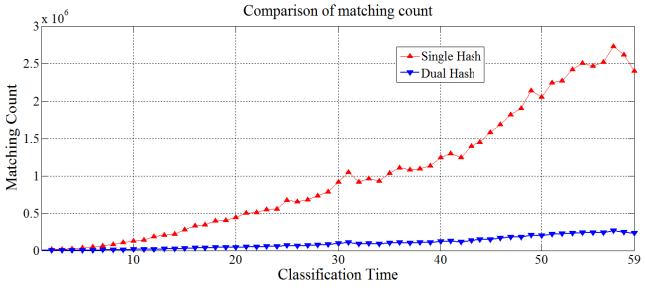

Figure 8. Comparison of Matching Count

We find that the dual hash could reduce comparison frequency more averagely decuple than single hash.

- *SSIP Cache Replacement*

Prompt replacement of SSIP cache entries is necessary in order to avoid explosion of the cached data. This happens because P2P applications generate a large amount of flows.

This study applies the RT-based (Registration Time based) and the RUT-based (Recently Used Time based) approaches for cache entry replacement. Both approaches are evaluated via experimental comparison on our campus traffic trace.

Figure 9 shows the results of the SSIP cache-based analysis that is used to evaluate the completeness of the RT-based and the RUT-based methods.

It shows that the completeness of the RUT-based method is higher than that of the RT-based method. This is because the RT-based deletion of the cache entry does not reflect recently used information. Hence, some flows cannot be classified until identified by the payload signature-based classification system.

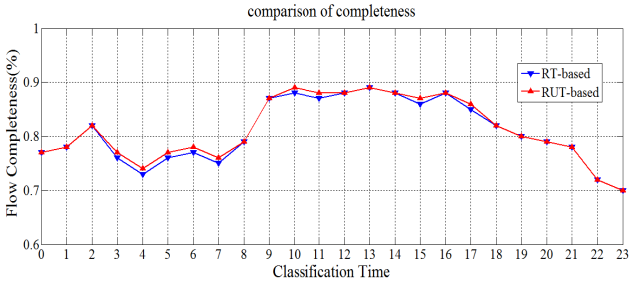

Figure 9. Comparison of Completeness

Figure 10 shows the results of the SSIP cache-based analysis that is used to evaluate the number of deletions and insertions in both the RT-based and the RUT-based methods. The number of replacements in the cache affects the processing time. The RT-based and RUT-based cache replacement methods registered an average of 73,743 and 53,214 records respectively.

The number of replacement operations is the same in both the RT-based and RUT-based methods when the classification system is operated for 3 hours, because there are only data insertions in the cache. After payload signature-based analysis, RT repeats the processing that deletes all the cache information and registers cache data.

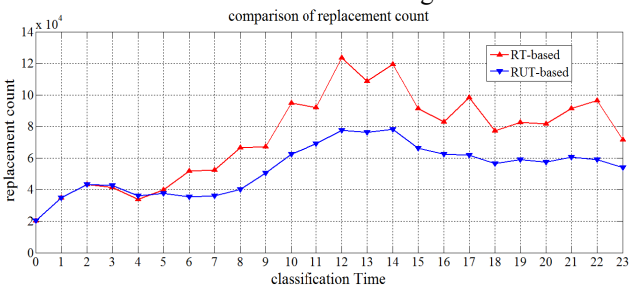

Figure 10. Comparison of Replacement Count

Considering the classification completeness and time, the RUT method is more effective than the RT method.

- *SSIP Cache Rearrangement*

In section 4.2, we proposed the dual hash memory structure to decrease collisions of hash key values. However, key value conflicts still occur frequently in the dual hash memory structure. Therefore, we propose a method to rearrange the cache data using the cache hit count. The cache data is sorted in descending order based on the hit count.

Figure 11 shows the total number of comparisons required to search the SSIP in the cache.

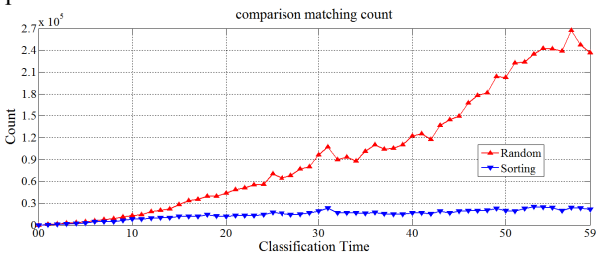

Figure 11. Comparison of Matching Count

It can be seen that when the alignment technique is applied, the maximum cache search space reduces by more than 13 times the number of matching attempts.

V. Evaluation

This section verifies the architecture of our traffic classification system. We built a baseline classification system to compare system performance.

Figure 12 shows the results of the signature-based analysis that was used to evaluate the processing speed of each method. We compared our proposed method to the baseline system. The suggested architecture achieved a nearly 10-fold improvement in processing speed over the payload signature-based classification system. Due to the nature of the campus network, the amount of traffic increases rapidly during working hours. However, the proposed method did not significantly increase the processing time during this period.

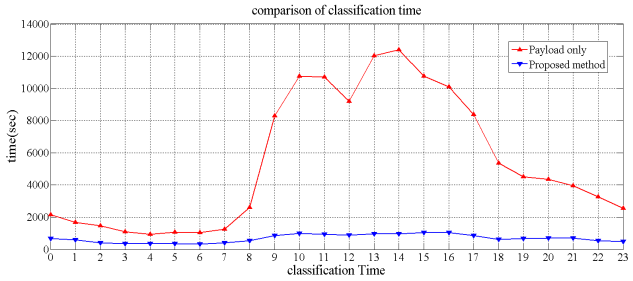

Figure 12. Comparison of Classification Time

Table 3 shows the results of the signature-based analysis used to evaluate the classification completeness of each method.

Table 3. Comparison of Completeness

Measure / Method		Flow		Packet		Byte	
		# of flow	Ratio	# of pkt	Ratio	# of Byte	Ratio
Total		51,477K	100.00%	2,012M	100.00%	1,578GB	100.00%
Payload	Classified	35,988K	69.91%	1,281M	63.69%	1,025GB	64.99%
	Unclassified	15,488K	30.09%	730M	36.31%	552GB	35.01%
Payload + SSIP Cache	Classified by payload	22,032K	42.80%	686M	34.10%	583GB	36.98%
	Classified by SSIP Cache	19,553K	37.98%	662M	32.93%	466GB	29.58%
	Unclassified	9,891K	19.22%	663M	32.97%	527GB	33.43%

We found that the proposed method improved the completeness in terms of flow by more than 10%. By means of the SSIP cache, it identified the flows that could not be classified via the payload signature-based classification.

VI. Conclusion and Future Work

The payload signature-based classification method shows the highest performance in terms of accuracy, completeness of flow classification, and practicality. However, this method has a significant drawback; it has a low processing speed when performing real-time processing in a high-speed network link. In this paper, we proposed a cache-based traffic classification method for improving the processing speed of the payload signature-based classification system.

We found that it is possible to design a high-speed Internet traffic classification system based on the proposed methods for cache-based classification. The suggested architecture achieved an approximately 10-fold improvement in processing speed over the payload signature-based classification system. In addition, it improved the completeness in terms of flow by more than 10%.

In this paper, we used the Recently Used Time-based cache replacement policy. In future, we plan to design a cache management method that takes into consideration a variety of factors such as traffic usage, communication type, and so on.

References

[1] J. S. Park, J. W. Park, S. H. Yoon, Y. S. Oh, M. S. Kim, "Development of Signature Generation System and Verification Network for Application Level Traffic Classification", in Proc. KIPS conf. Apr. 23-24, 2009, pp. 1288-1291, Pusan, Korea.

[2] S. H. Yoon, H. G. Roh, M. S. Kim, "Internet Application Traffic Classification using Traffic Measurement Agent", in Proc. KICS Jul. 2-4, 2008, pp. 618, Jeju Island, Korea.

[3] A. Dainotti, A. Pescape, K. Claffy, Issues and future directions in traffic classification, IEEE Network: The Magazine of Global Internetworking, Vol. 26, 2012, pp. 35-40.

[4] K. Xu, M. Zhang, M. Ye, M. Chiu, J. Wu, Identify P2P traffic by inspecting data transfer behaviour, Computer Communications, Vol. 33, 2010, pp. 1141-1150.

[5] F. Risso, M. Baldi, O. Morandi, A. Baldini, and P. Monclus, "Lightweight, Payload-Based Traffic Classification An Experimental Evaluation," IEEE International Conference on Communications, Beijing, China, May. 19-23, 2008, pp. 5869-5875.

[6] S. H. Yoon, J. W. Park, Y. S. Oh, J. S. Park, and M. S. Kim, "Internet Application Traffic Classification Using Fixed IP-port," APNOMS 2009, LNCS, Sep. 23-25, 2009, pp. 21-30, Jeju, Korea.

[7] F. Yu, Z. Chen, Y. Dino, T. V. Lakshman, R. H. Katz, "Fast and Memory Efficient Regular Expression Matching for Deep Packet Inspection", ANCS 2006, Dec. 2006, San Jose, California, USA.

[8] C. L. Hayes, Y. Luo, "DPICO: a High Speed Deep Packet Inspection Engine Using Compact Finite Automata", ACM/IEEE Symposium on Architecture for Networking and Communications Dystems, Dec. 03-04, 2007, Orlando, Florida, USA.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Chapter 32: String Matching, pp. 906–932.

[10] A. Mitra, W. Najjar, L. Bhuyan, "Compiling PCRE to FPGA for Accelerating SNORT IDS", Proceedings of the 3rd ACM/IEEE Symposium on Architecture for networking and communications systems, Dec. 03-04, 2007, Orlando, Florida, USA.

[11] T. Y. Park, S. H. Yoon, M. S. Kim, "A Study on a Resolution of the Massive Using the Dual Hash Structure", KNOM 2013, May. 09-10, 2013, pp. 112-116, Daegu, Korea.