Journal of Theoretical and Applied Information Technology
30th September 2020. Vol.98. No 18

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ANALYSIS OF FLOW ENTRY REPLACEMENT ALGORITHMS IN INTELLIGENT NETWORKS

[1]HYONGYOUNG KO [1]YEHOON JANG [1]NAMGI KIM
[1]Department of Computer Science and Engineering, Kyonggi University, Suwon 16277
E-mail: {kgu15460, jyh94, ngkim}@kyonggi.ac.kr
*Corresponding Author: Namgi Kim (ngkim@ kyonggi.ac.kr)

ABSTRACT

In recent years, as Internet services have diversified and grown rapidly, intelligent network environment has dynamically changed with parameters such as traffic patterns and network topologies. To flexibly manage such dynamic changes, SDN (Software Defined Network) technology has emerged. SDN enables more flexible Internet services by dividing the network architecture into a control plane and data plane. By the way, in the SDN, a problem with flow entry replacement may arise owing to flow table size restrictions within the switches. A flow entry replacement problem can increase the packet processing time and degrade the quality of service for the users. Therefore, we need to know exactly the performance of the flow entry replacement algorithms. In order to practically analyze the performance of the flow entry replacement algorithms, we first collect and analyze the actual Internet traffics of famous Internet services such as Instagram, Facebook, Youtube, and Netflex. Then, we analyze the performances of flow entry replacement algorithms by the collected traffic data. Based on the results, the LFU (Least Frequently Used) algorithm exhibits the worst performance, whereas the FIFO (First In First Out), LRU (Least Recently Used), and SFF (Short Flow First) algorithms show relatively better performances.

Keywords: *Intelligent Network, Internet traffic, SDN, Flow Entry Replacement, Flow Table Management*

1. INTRODUCTION

With the emergence and development of various Internet services, Internet traffic patterns have become more diverse and complex. These changes may increase the network overhead and degrade the quality of service for users. To solve such problems, data centers have been established to perform the distributed processing of traffic. However, there are limitations in solving these problems owing to the closed network architectures of traditional Internet. To reflect new Internet services and traffic patterns, traditional network architectures need to perform updates on each device through separate independent accesses. Further, some functions may not perform properly owing to compatibility issues. To fundamentally address such issues, SDN (Software Defined Network) technology has been developed. The SDN features a network virtualization approach in optimizing network resources and flexibly adapting to network requirement and traffic changes [1].

The SDN divides the network architecture into the two planes: control plane and data plane. The control plane performs bandwidth management, forwarding control, resource management, etc. based on the logically centralized network topology. As a result, the SDN administrator can identify and optimize the network from a global perspective according to the operating environment. In addition, the SDN can determine the differentiated forwarding and packet processing for operation according to the given policy. The SDN switches are executed by receiving commands from the SDN controller. As a result, packets are simply forwarded in the switches without executing the complex functions in data plane [2][3].

In the SDN, packets are managed by units of flow. Flows are generated in the SDN controller based on the information such as the packet source and destination. The flow tables, which have information about flows, are managed and stored in the SDN switches. Once a packet is arrived, the switch searches a flow entry in the flow table corresponding to the packet. If a matched flow entry is found, packet processing, such as packet delivery, modify, and destroy, is executed using the information in the entry. If there is no entry that matches the packet, the SDN switch requests the SDN controller for generating a flow of the corresponding packet. After receiving the flow

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

information from the controller, the switch stores it in the flow table and processes the packet according to the flow information.

In a large-scale network architecture, not all network devices can be homogenous owing to the needs or costs of each device function. Therefore, heterogeneous SDN switches are generally employed and the different switches which have the different size of the flow table experience different flow entry replacements. The frequent flow entry replacement may increase the number of exchanged messages with the controller and delay in packet delivery [4][5][6]. Therefore, we need to have an efficient flow entry replacement algorithm which reduces the number of the replacement and increases the flow entry matching rate for improving the network performance. This is the research question of this paper.

In order to justify the need of the flow entry replacement algorithms in practical, we first collect and analyze the actual Internet traffic data of famous Internet services such as Instagram, Facebook, Youtube, and Netflex. Then, we analyze the performance of the representative flow entry replacement algorithms such as FIFO (First In First Out), LFU (Least Frequently Used), LRU (Least Recently Used), and SFF (Short Flow First) [7] by the collected Internet traffic data.

2. RELATED WORK

In the SDN network, the switch executes only the simple forwarding functions. Instead, the controller manages the network and sends flow information for the packet processing to the switches. Therefore, it is very important to reduce the controller overhead for improving the network performance. In order to reduce the controller overhead, in [8], the switch that is processing a large amount of traffic is migrated to another controller that is processing a relatively smaller amount of traffic. However, in this method, the controller overhead may increase owing to additional message exchanges for switching the controller. As another method of reducing the overhead, in [9], the Kandoo framework is proposed to divide the controller into two levels: root and local. This method involves the root controller processing the requests for the entire network status, while the remaining requests are processed by the local controllers. In [10], a tag-in-tag method was proposed to reduce the flow entry by delivering flow to another switch. Similar to [10], in [11], a method was proposed to classify the network into core and edge. It adopts ARP (Address Resolution Protocol) spoofing using the virtual MAC (Media Access Control) address and physical MAC address. However, this method is hard to implement as ARP spoofing requires changes of the kernel implementation of the host.

The related works discussed above have presented methods such as changing the network architecture or forwarding traffic as a means of reducing network overhead. However, not many studies have been conducted for solving the overhead that can occur when the flow entry needs a replacement owing to a full flow table. In [7], the SFF (Short Flow First) algorithm was proposed to select a target flow entry for replacement based on the flow characteristics when the flow table is fully occupied. However, synthetic Internet traffic data, which were manually generated by the authors of that study, were used for the performance evaluation. The performance of the SFF algorithm in the real Internet traffic environment remains uncertain.

In this paper, we first analyze the actual Internet traffic patterns by collecting them from the popular Internet services. Then, we evaluate the performances of the representative algorithms that can be used for flow entry replacement. These are the contributions of our paper.

3. TRAFFIC COLLECTION AND ANALYSIS OF REPRESENTATIVE INTERNET SERVICES

Table 1: Environments for traffic collection

Parameters	Values
OS	Ubuntu 16.04
Browser	Chrome v.77.0.3865.75
Packet capture tool	Wireshark v.2.6.8

Table 2: Internet services for traffic collection

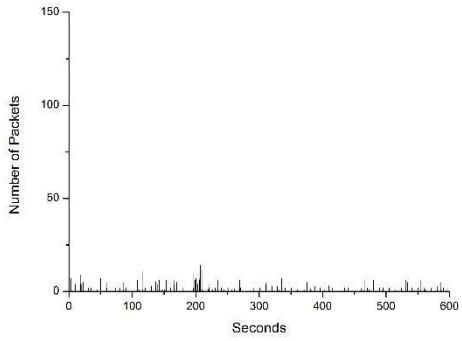
Services	Contents
Instagram	Social network data (video, photo, text, etc)
Facebook	Social network data (video, photo, text, etc)
Youtube	Stored multimedia data (video, audio)
Netflix	Stored multimedia data (video, audio)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

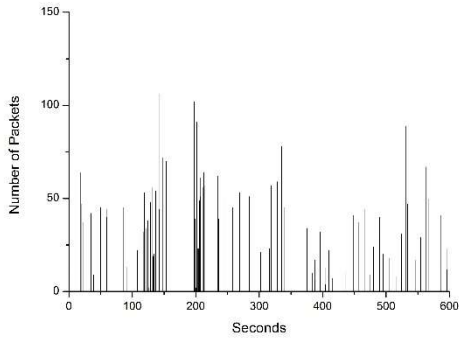
In this section, we show the actual Internet traffic data that were collected and analyzed. Table 1 shows the traffic capture environment and Table 2 shows the list of Internet services and content types for collecting traffic data.

3.1 Instagram traffic

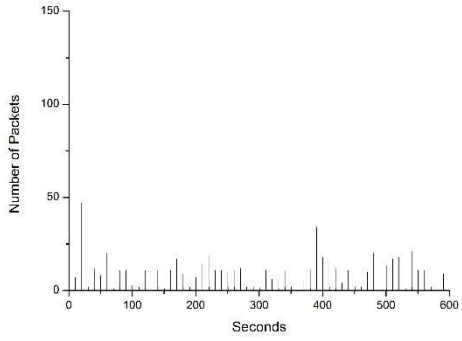
Instagram [12] is designed to receive content when a user requests content information. The incoming information includes the content uploader's user name, profile picture, content photo, and profile pictures of the users who left comments along with comment information. As shown in Figure 1, Instagram service receives data in three main flows: query, content, and log flows. The query flow indicates the requested content information by the user and the request result. The content flow is for sending content and the log flow collects logs for optimizing the user environment.



(a) Query flow

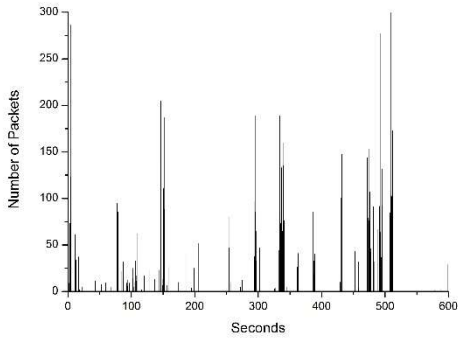


(b) Content flow

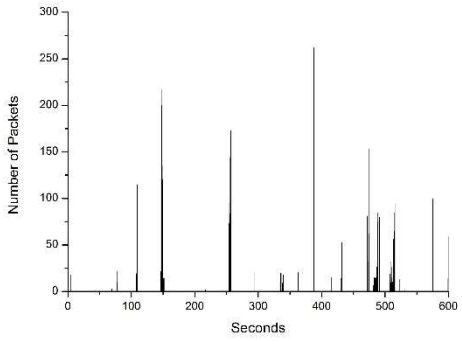


(c) Log flow

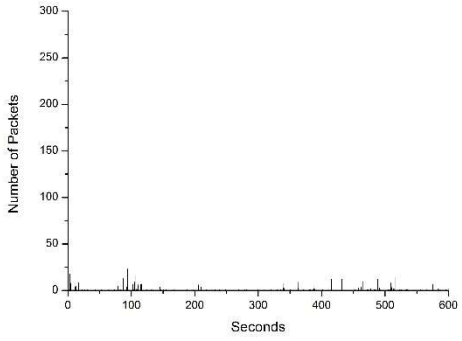
Figure 1: Traffic pattern of Instagram service



(a) Picture traffic



(b) Video traffic



(c) Log, query, and module traffic

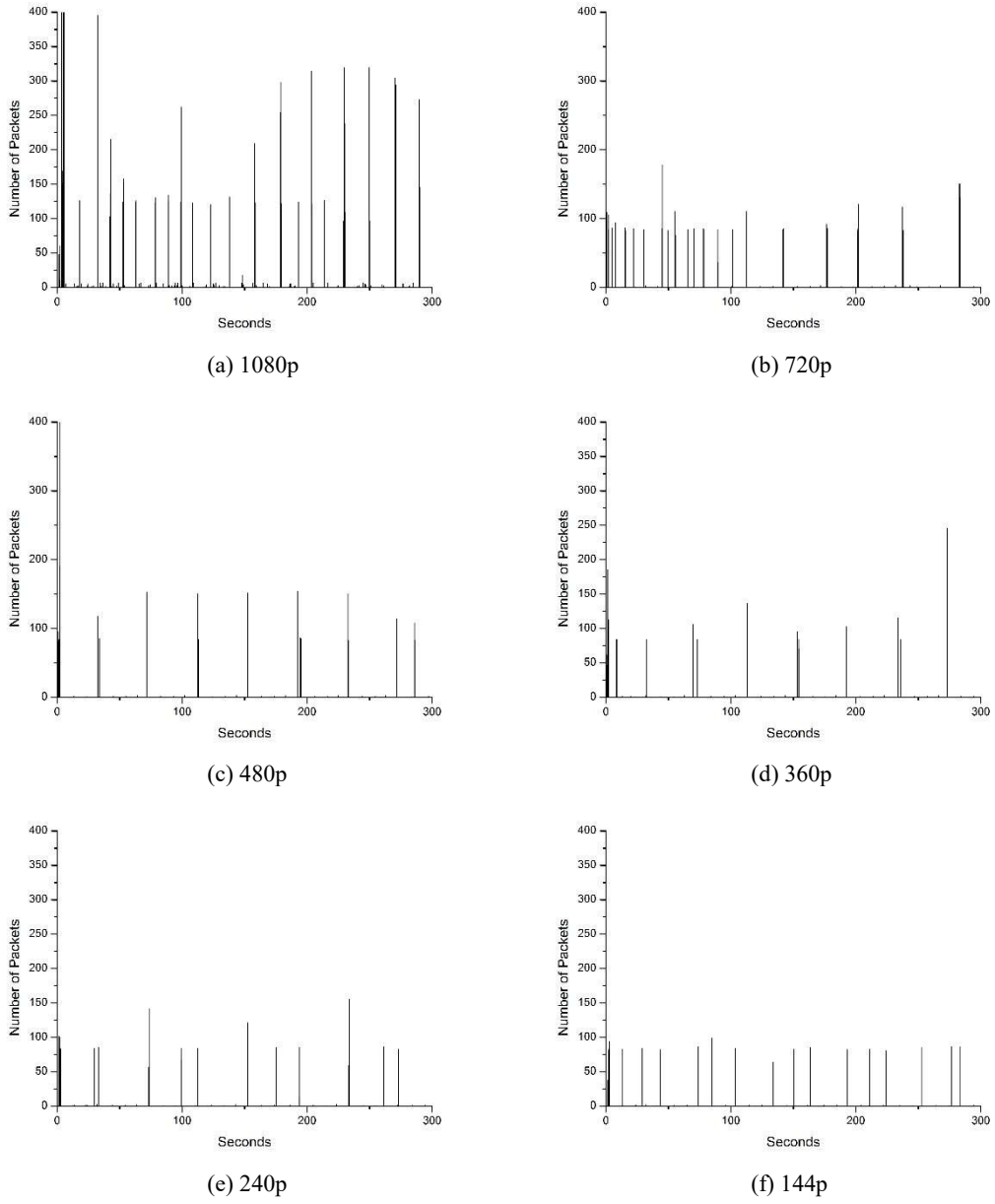
Figure 2: Traffic pattern of Facebook service

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

As shown in the figure, the packets are periodically transmitted every 10 seconds in the log flow even if the content does not include any video data.

3.2 Facebook traffic

Facebook [13] receives relatively more information than Instagram. Facebook receives not only requested contents but also various types of additional data such as story photos, streaming, and advertisements. As shown in Figure 2, Facebook also transmits data in three main flows. The picture traffic flow indicates the Cascading Style Sheets and JavaScript files related to photos or user interfaces. The video traffic flow indicates the flow related to video data. Lastly, the log, query, and module traffic flow indicates the flow related to log, query, and module call data. Similar to Instagram, in the content, query, and module call flow, packets are transmitted irregularly as the data are processed according to the user requests. However, the



(a) 1080p (b) 720p

(c) 480p (d) 360p

(e) 240p (f) 144p

Figure 3: Traffic pattern of Youtube service

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

packets related to logs are periodically transmitted every 5 seconds.

3.3 Youtube traffic

Youtube [14] is the largest video sharing Internet service in the world, enabling any user to watch, upload, and share videos. Youtube allows users to watch videos in different qualities such as 144p, 240p, 360p, 480p, 720p, and 1080p. The Uniform Resource Locator (URL) for Youtube traffic collection is shown in Table 3.

Table 3: Youtube content URL for traffic collection

URL	Content Type
https://youtu.be/kQJ4ulRELHo	video

The traffic patterns collected from the Youtube service are displayed in Figure 3. As shown in the figure, the packets are received at regular intervals in each video quality. The data transmitting intervals become shorter and the number of packets gradually increases when the video quality is higher.

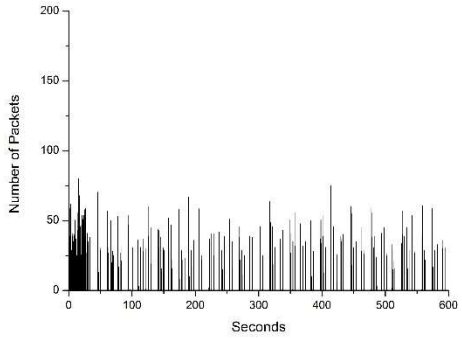
3.4 Netflix traffic

Netflix [15] is a service that shares pre-produced multimedia contents. Netflix supports the streaming of videos in three different qualities: low quality, medium quality, and high quality. The low quality video generates a data transfer rate of up to 0.3 GB per hour. The medium quality video generates up to 0.7 GB per hour. Lastly, the high quality video generates up to 3 GB for HD, and up to 7 GB for 4K UHD resolutions per hour. The content used for collecting Netflix traffic is shown in Table 4.

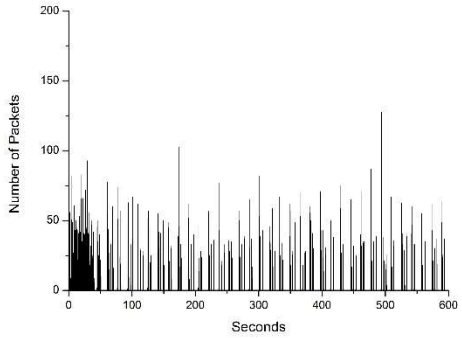
Table 4: Netflix content for traffic collection

Content	Type
Ingress: the animation	animation

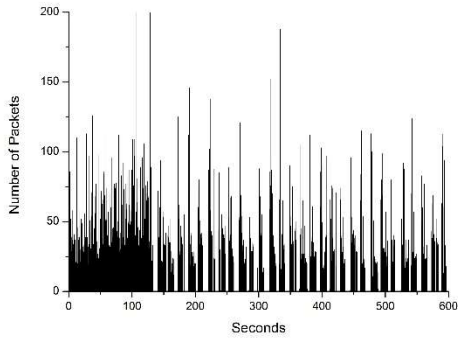
The collected Netflix traffic results are displayed in Figure 4. In the Netflix traffic, data transmission intervals were almost the same for all video qualities. Instead, the number of packets is increased when the requested video quality is high.



(a) Low quality video



(b) Middle quality video



(c) High quality video

Figure 4: Traffic pattern of Netflix service

3.5 Traffic pattern summary

Although there exist a small portion of video data, Instagram and Facebook data mostly comprise web data. Nevertheless, a periodic packet transfer was observed owing to the log collections for user optimization. Youtube and Netflix use the MPEG-DASH (Moving Picture Expert Group-Dynamic Adaptive Streaming over HTTP) [16] protocol. Whereas Netflix frequently forwards packets in short interval periods, Youtube has a

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

traffic pattern that collects and transfers large chunks of packets over longer intervals.

4. REPRESENTATIVE FLOW ENTRY REPLACEMENT ALGORITHMS

The representative algorithms that can be used for replacing flow entries in the flow table of the SDN switch include the widely known FIFO, LFU, and LRU algorithms along with the SFF algorithm. The FIFO algorithm replaces the entry stored for the longest time in the flow table. The concept of the algorithm is to replace the entry that has been retained for the longest time when adding a new entry as the old entry has been sufficiently used. The FIFO algorithm method is inefficient when there is a large flow of packet transmission for a long time as it selects the target entry for deletion based on the duration of registered time

The LFU algorithm replaces the entry that has the lowest matching number among the entries in the flow table. The concept here is that an entry with a high number of matches is likely to have more matches in the future as well. If a recently added entry may no longer be used when the flow table is full, the LFU algorithm can continuously perform at a high level. However, it can become inefficient once there is a large flow of continuous packet transmission.

The LRU algorithm replaces the entry that has not been matched for the longest time in the flow table. The algorithm can perform well under the assumption that the entry that has not been matched for the longest time will no longer be used. In addition, it has the advantage that the recently matched entry is retained in the flow table regardless of when the flow is registered or the number of times that the packets have used the entry.

The SFF algorithm categorizes a flow entry into a short flow or long flow based on the flow matching cycle. It assumes that the packets of the short flow are transmitted for a short period of time and the packets of the long flow are continuously transmitted for a long period of time. Therefore, this algorithm attempts to replace the entry that has not been matched for the longest time among the short flow entries first. If no corresponding entry is found among the short flow entries, it selects the target entry for deletion from the long flow entries. A typical Internet service example of a short flow is a web service and an example of a long flow is a streaming service.

5. PERFORMANCE EVALUATION WITH ACTUAL INTERNET TRAFFICS

To evaluate the performances of the flow entry replacement algorithms, an experimental network topology was constructed using Mininet emulator [17], as shown in Figure 5. The topology was designed to have the packets start from four web server and four video servers, and arrive at 40 client nodes.

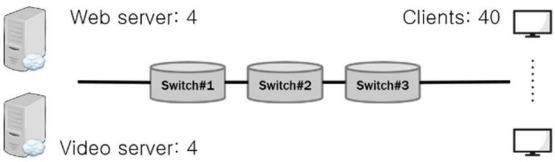


Figure 5: Experimental topology

The packets used in the experiment were generated through the tcpreplay packet tool [18] based on the traffic patterns collected from the actual Internet. The flow table size of the SDN switch used in the experiment varied from 10 to 30 entries. The details of the experimental parameters are summarized in Table 5.

Table 5: Experimental parameters

Parameters	Values
Flow table size (SW1-SW2-SW3)	10-10-10, 20-20-20, 30-30-30 10-20-30, 30-20-10, 30-10-30, 10-30-10
Number of flows	40 web and 40 video flows
Number of packets	830K

5.1 Flow miss rate

A flow miss occurs when an incoming packet does not match any existing flow entry in the flow table. When such a flow miss occurs, the switch requests the controller to generate a new flow. It results in message overhead owing to multiple message exchanges between the controller and switch. The flow miss rate indicates the rate of packets having flow misses out of the entire group of packets. If the flow miss rate is high, the message overhead between the switch and flow increases. Frequent packet delays also occur, where the packets are blocked and cannot be forwarded until a new flow entry is generated.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 6 shows the flow miss rates with a flow table size of 10 flow entries in switches #1, #2, and #3. As shown in the figure, the LFU algorithm exhibited the highest flow miss rate, whereas the FIFO, LRU, and SFF algorithms demonstrated relatively lower flow miss rates. The same flow table sizes were used in switches #1, #2, and #3 and the number of flow misses was almost the same in each switch.

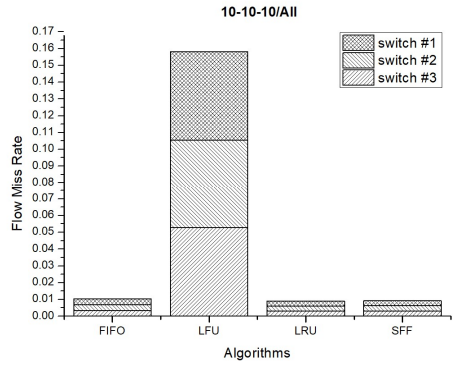


Figure 6: Flow miss rate with 10-10-10 flow entries

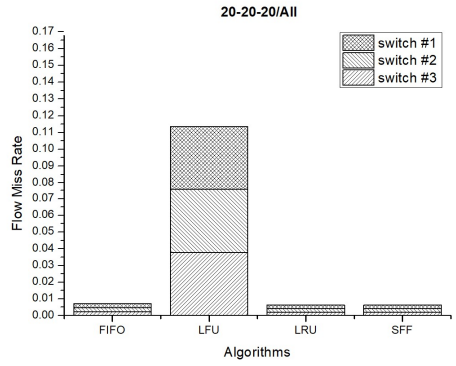


Figure 7: Flow miss rate with 20-20-20 flow entries

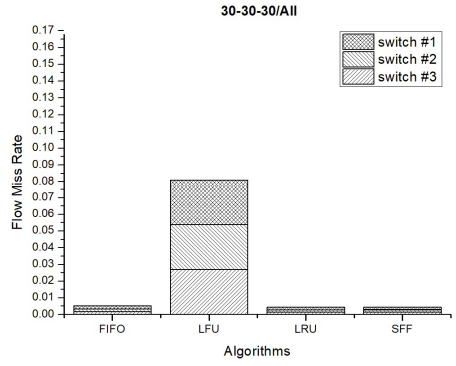


Figure 8: Flow miss rate with 30-30-30 flow entries

Figures 7 and 8 display the miss rates when the flow table sizes in switches #1, #2, and #3 are 20 and 30, respectively. As observed from the results, the flow miss rates decreased with the increase in the flow table size owing to the increased probability of having the flow corresponding to the incoming packet.

Figures 9, 10, 11, and 12 display the experiment results of configuring different flow table sizes for each switch. Figure 9 shows the flow miss rates of each algorithm when the flow table sizes are 10, 20, and 30 for switches #1, #2, and #3, respectively. As shown in the figure, switch #1 had the highest number of flow missed packets. This is because the flow table size of switch #1 was the smallest.

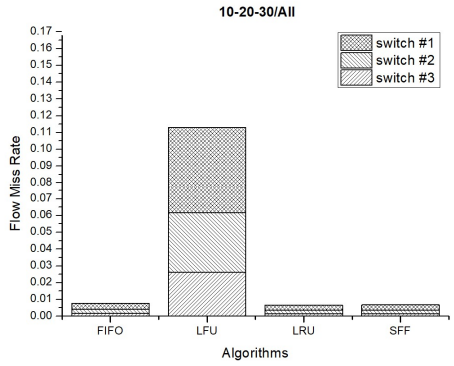


Figure 9: Flow miss rate with 10-20-30 flow entries

Figure 10 shows the flow miss rates of each algorithm when the flow table sizes are 30, 20, and 10 for switches #1, #2, and #3, respectively. Here, the highest flow miss rate was found in switch #3 because of the switch having the smallest flow table size.

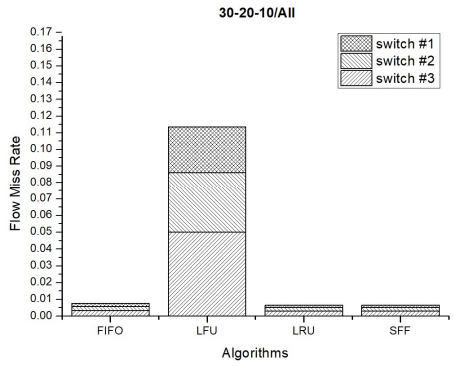


Figure 10: Flow miss rate with 30-20-10 flow entries

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 11 shows the flow miss rates of each algorithm when the flow table sizes are 30, 10, and 30 for switches #1, #2, and #3, respectively. Here, the highest flow miss rate was found in switch #2 because it had the smallest flow table size.

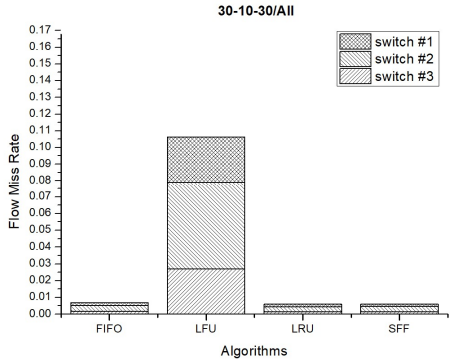


Figure 11: Flow miss rate with 30-10-30 flow entries

Figure 12 shows the flow miss rates of each algorithm when the flow table sizes are 10, 30, and 10 for switches #1, #2, and #3, respectively. Here, switches #1 and #3 had the highest flow miss rates as they had the smallest flow table sizes at 10, and switch #2 had almost negligible flow misses.

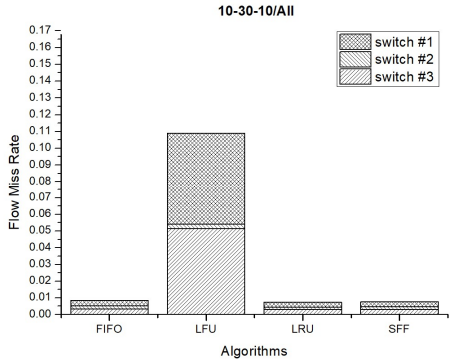


Figure 12: Flow miss rate with 10-30-10 flow entries

Thus, the flow miss rates of the flow entry replacement algorithms have been measured using actual Internet traffic patterns in the environment of having various flow table sizes. The experiment results showed the highest flow miss rates in the LFU algorithm in all cases. In addition, FIFO, LRU, and SFF algorithms showed relatively lower flow miss rates compared with the LFU algorithm.

5.2 Packet delay due to flow entry replacement

If an incoming packet enters the switch and the corresponding flow entry is not in the flow table, the switch asks the controller to generate a flow. During this process, a delay is occurred and the packet cannot be forwarded to the next switch until the new flow entry is created in the switch. This type of delay can cause quality degradation of the services to which the corresponding packet belongs. Thus, it can be inferred that the service qualities improve with the decrease in the packet delays owing to the flow misses.

Figure 13 shows the delayed packet ratio in the switch when the flow table size is 10 for switches #1, #2, and #3. If the ratio of delayed packets displays 1, it denotes that all the packets transmitted within the network have been delayed owing to the flow miss in all switches. The legend 111 shown in the figure indicates the switch number in which the packet has been delayed. 111 represents the packets that delay in switches #1, #2, and #3. In the same way, 100 represents the packets that delay in switches #1 only. As seen in the figure, the LFU algorithm had the largest number of delayed packets. On contrary, the FIFO, LRU, and SFF algorithms had a relatively lower number of delayed packets.

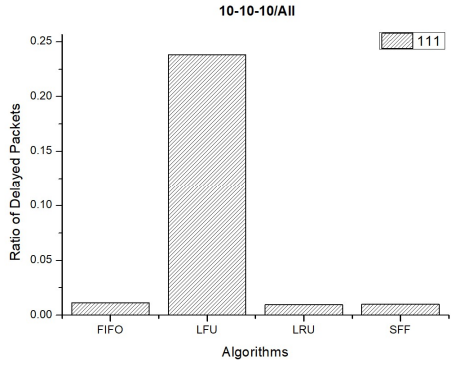


Figure 13: D*elayed packets with 10-10-10 flow entries*

Figures 14 and 15 show the delayed packet ratios of each algorithm when the flow table sizes of switches #1, #2, and #3 are 20 and 30 flow entries, respectively. As shown in the figures, the number of delayed packets decreases with the increase in the flow table size. This is owing to the increase in the number of flow entries that can be stored in the table with the increase in the flow table size; this leads to a lower number of flow misses. In addition, all the packets that have been delayed in switch #1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

are also delayed in switch #2 and switch #3. This result is obtained because of the flow table sizes being the same in all three switches.

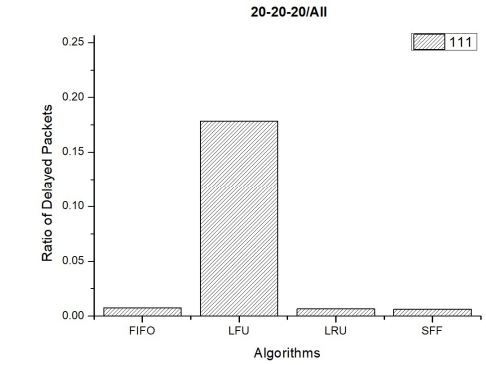


Figure 14: Delayed packets with 20-20-20 flow entries

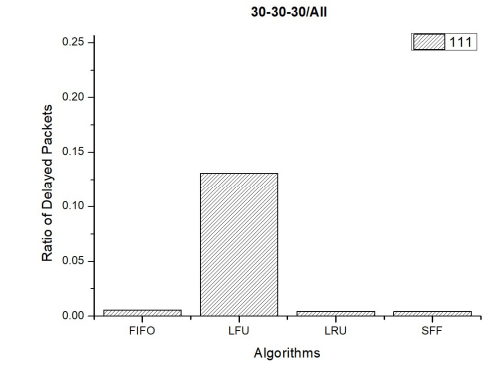


Figure 15: Delayed packets with 30-30-30 flow entries

Figure 16 shows the delayed packet ratios of each algorithm when the flow table sizes are 10, 20, and 30 for switches #1, #2, and #3, respectively. As observed from the results, switch #1 had the most packet delays because switch #1 had the smallest flow table size.

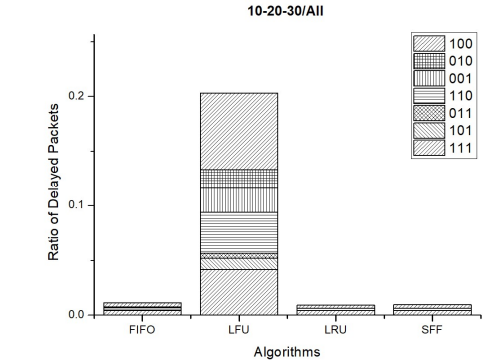


Figure 16: Delayed packets with 10-20-30 flow entries

Figure 17 shows the delayed packet ratios of each algorithm when the flow table sizes are 30, 20, and 10 for switches #1, #2, and #3, respectively. Here, switch #3 had the most packet delays as it had the smallest flow table size.

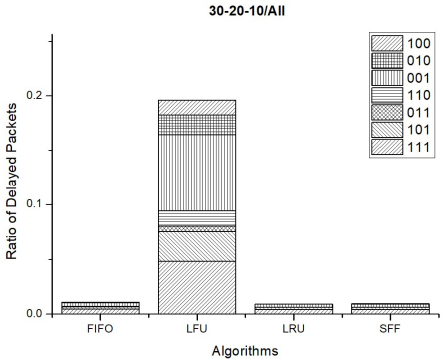


Figure 17: Delayed packets with 30-20-10 flow entries

Figure 18 shows the delayed packet ratios of each algorithm when the flow table sizes are 30, 10, and 30 for switches #1, #2, and #3, respectively. Here, switch #2 had the most packet delays as it had the smallest flow table size.

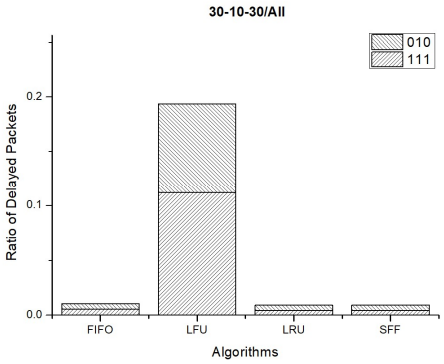


Figure 18: Delayed packets with 30-10-30 flow entries

Figure 19 shows the delayed packet ratios of each algorithm when the flow table sizes are 10, 30, and 10 for switches #1, #2, and #3, respectively. Here, the flow information in the flow table of switch #1 and that of switch #2 become different as the flow table size of switch #1 is smaller than that of switch #2. Therefore, most packet delays are found either only in switch #2 or in all three switches.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

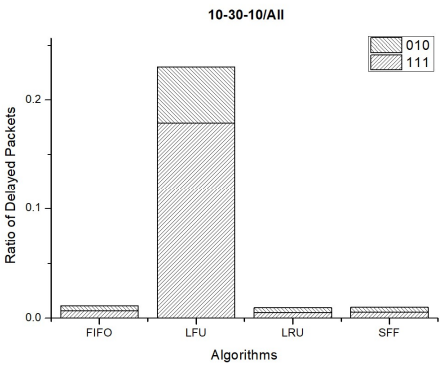


Figure 19: Delayed packets with 10-30-10 flow entries

Until now, the packet delay ratios have been measured using actual Internet traffic patterns in environments having various flow table sizes. The experiment results showed the highest packet delay ratios in the LFU algorithm in all cases. In addition, compared with the LFU algorithm, the FIFO, LRU, and SFF algorithms showed that relatively smaller number of packets have delays owing to flow misses.

6. CONCLUSIONS

In this study, the research objectives are to analyze the actual Internet traffic patterns from the popular Internet services and evaluate the performances of the flow entry replacement algorithms that can be used in the SDN environment using them. The Internet traffic analysis results showed that most traffic data has no specific periodicity and has short transmission times. However, some traffic data are periodically transmitted in the process of collecting logs to optimize user environment or to provide advertisement information. Further, video service traffic data, such as Youtube and Netflix, showed different traffic patterns. However, all video service traffic had the same pattern that increases the number of packets and shortens the packet interval as video quality increases.

In this study, the performances of the FIFO, LFU, LRU, and SFF algorithms, which can be used in flow entry replacement, were evaluated using actual traffic data collected from the Internet. Based on the results, the LFU algorithm displayed the worst performance owing to the frequent replacements. The LRU and SFF algorithms displayed relatively better performances. In [7], it was proposed that the SFF algorithm performed better than the LRU algorithm. However, in this study, the performances of the SFF and LRU were similar because the flows within the traffic were mostly uninterrupted and maintained for the entire transmission time. Since the traffic used in the experiment was maintained until most of the flows ended, the SFF algorithm stores these continuously matched flows as long flows and long flow entries increased whereas the short flow entries decreased. As a result, the performance of the SFF algorithm was similar to that of the LRU algorithm in the flow entry replacement process.

The limitation of this study is that the collected data from the Internet are a little limited. In this work, we collected actual Internet traffic from the five popular Internet services that are representative Internet services. However, they may not capture all features of the Internet traffic. In order to solve this problem, we plan to collect more diverse Internet traffic data and analyze the algorithm performances in a more realistic Internet environment to develop a better performing flow entry replacement algorithm.

ACKNOWLEDGMENTS:

This work was supported by Kyonggi University Research Grant 2019.

REFRENCES:

[1] R. Amin, M. Reisslein, and S. Nadir, "Hybrid SDN networks: A survey of existing approaches," IEEE Communications Surveys & Tutorials, Vol. 20, No. 4, pp.3259-3306, 2018.

[2] Open Networking Fundation, "Software-Defined Networking: The new norm for networks," ONF White Paper, Vol. 2, No. 11, pp. 2-6, April 2012.

[3] S. Sezer, et al, "Are we ready for SDN? Implementation challenges for software-defined networks," IEEE Communication Magazine, Vol. 51, No. 7, pp. 36-43, July 2013.

[4] P. Dely, et al, "Openflow for wireless mesh network," IEEE Computer Communications and Networks, pp. 1-6, 2011.

[5] S. Kandula, et al., "The nature of data center traffic: measurements & analysis," ACM SIGCOMM conference on Internet measurement, pp. 202-208, 2009.

[6] T. Benson, et al., "Network traffic characteristics of data centers in the wild," ACM SIGCOMM conference on Internet measurement, pp. 267-280, 2010.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[7] N. Kim, et al., "A new flow entry replacement scheme considering traffic characteristics in Software-Defined Networks," Applied Sciences, Vol. 10, No. 10, pp. 3590, 2020.

[8] A. Dixit, et al., "Towards an elastic distributed SDN controller," ACM SIGCOMM workshop on Hot topics in software defined networking, Vol. 43, No. 4, pp. 7-12, 2013.

[9] S. Hassas Yeganeh and Y. Ganjali., "Kandoo: a framework for efficient and scalable offloading of control applications," The 1st workshop on Hot topics in software defined networks, pp. 19-24, August 2012.

[10] S. Banerjee and K. Kalapriha, "Tag-in-tag: Efficient flow table management in SDN switches," 10th IEEE International Conference on Network and Service Management (CNSM), pp. 109-117, Nov. 2014.

[11] K. Agarwal, et al., "Shadow MACs: Scalable label-switching for commodity ethernet," The 3rd workshop on Hot topics in software defined networking, pp. 157-162, 2014.

[12] Instagram, https://www.instagram.com/ [Accessed on April 1, 2020]

[13] Facebook, https://www.facebook.com/ [Accessed on April 1, 2020]

[14] Youtube, https://www.youtube.com/ [Accessed on April 1, 2020]

[15] Netflex, https://www.netflix.com/ [Accessed on April 1, 2020]

[16] MPEG-DASH, "Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats", ISO/IEC 23009-1:2012, Apr. 2012.

[17] Mininet, http://mininet.org/ [Accessed on April 1, 2020]

[18] tcpreplay (Replay captured network traffic), http://tcpreplay.appneta.com/ [Accessed on April 1, 2020]