Traffic-Aware Networking for Video Streaming Service Using SDN

Calvin Hue*, Yu-Jia Chen† and Li-Chun Wang†
*Department of Photonics, National Chiao Tung University, Hsinchu, Taiwan
†Institute of Electrical Engineering, National Chiao Tung University, Hsinchu, Taiwan
Email: cal.eo00@nctu.edu.tw, allan920693.cm99g@nctu.edu.tw and lichun@cc.nctu.edu.tw

***Abstract*—In this paper we propose a traffic-aware networking technique by using software-defined networking (SDN) to precisely and promptly identify video streaming packets. We performed experiments on our SDN testbed. Compared with the existing deep packet inspection (DPI) method, the proposed SDN-enabled traffic-aware packet routing technique can reduce the latency by 75% and increase the success rate for traffic identification up to 138% .**

Keywords—Traffic-aware Networking, Software Defined Network, Deep Packet Inspection.

I. Introduction

Internet traffic has grown more than fivefold in the past five years, and the trend of growth is estimated to continue for the next five years [1]. Traffic capacity is highly related to the underlying network structure and routing strategies[2]. Therefore, a new network structure combined with an intelligent routing strategy for increasing traffic capacity is of foremost importance.

Deep packet inspection (DPI) is a popular filtering technique examining the data (and header) part of a packet, which can be used to identify the content type of packet (e.g., video, text). The results of DPI can be exploited by an application-aware network to effectuate smart routing strategies in the network and further improve network users' quality of experience (QoE), which is a metric to evaluate application performance. However, this technique has significant disadvantage, for example, it is time-consuming and energy-intensive.

The rise of Software-Defined Network (SDN) is a chance for researchers to resolve the foregoing network problem. SDN is a new network architecture that decouples the control plane from the data plane. In actual implementation, a centralized controller having a global view of the entire network is designed. This design makes network behavior programmable and management flexible. Studies have been undertaken to use SDN to improve the QoE for network users while they are streaming videos.[3][4]

The goal of this study is to promptly identify video streaming traffic without using DPI, so we proposed an inspection-free traffic-aware networking technique leveraging the statistics information and characteristics of SDN. Different from DPI, the proposed technique uses the statistics of flows in SDN to identify the flows' content type and this technique is not energy-intensive and processor-intensive. In addition, the new technique also requires less knowledge and operating cost for deployment. To demonstrate the utility of the proposed technique, the traffic data of YouTube videos was analyzed, and the response time, success rate and false positive rate to identify video streaming packets were collected in the study. Contributions and novelties of this work are listed as follows:

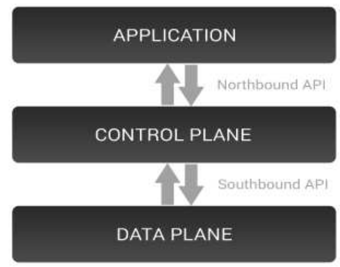

Fig. 1. The three-layered architecture of SDN.

- Opposed to DPI, a technique was proposed which identifies video streaming flows by analyzing traffic pattern without inspecting the packets.
- The proposed technique leverages the statistics data natively provided by SDN without using third-party extensions, so it can be easily extended and deployed in the network.
- The technique has been tested in real SDN testbed. Experimental results show that the proposed technique is capable of reducing the latency by 75% or increasing the success rate up to 138% for traffic identification compared with the existing method using DPI.[3] The trade-offs between the statistics retrieval time, latency and the success rate were observed and discussed in this study. We also compared how the proposed technique responds to traffics generated by other user activities.

In the remaining part of this paper, background information about SDN, methods to identify video traffic and traffic characteristics of video streaming service will be provided. Then the system scenario of this study will be introduced, following by explaining the mechanism of our traffic-aware networking technique. Finally, the testbed setup and experimental results are presented and discussed.

978-1-4673-8590-9/15/$31.00 ©2015 IEEE

II. Background and Related Work

A. Software-Defined Networking

SDN attracts researchers' attention because of the soaring needs for Internet capacity and the inconvenience caused by the current network infrastructure. The origin of SDN can be dated back to 1995.[5] The concept is to provide the network administrator centralized and modifiable control of the traffic in the network.

In implementation, the architecture of SDN can be seen as consisting of three layers — application, the control plane and the data plane as illustrated in Fig. 1. Between application and the control plane, there exists a programmatic interface called the Northbound API, which enables applications to be executed on top of the network. Northbound API is valuable because it makes the controller programmable and responding to changes in the network. In addition to the Northbound API, there is the Southbound API that bridges the data plane and the control plane. SDN architecture segregates control planes from forwarding devices. The segregated control planes are combined in design and therefore regarded as a centralized controller which has a global view of the entire network. Hence, flexible reconfiguration of the network becomes possible by the introduction of a centralized controller. In 2008, the protocol OpenFlow was proposed and studied in many studies.[6] By promotion from the Open Networking Foundation, OpenFlow has become the standard for realizing the Southbound API. OpenFlow switches report statistics of flows to the controller with help from the OpenFlow protocol. Therefore, researchers and developers can easily access the statistics in applications through the Northbound API. These statistics are invaluable, since they reflect the traffic situation within the network.

B. Identifying Video Streaming Service

Video streaming service has a phenomenal growth in recent years. According to the study from Cisco[1], one million minutes video content is predicted to transmit across the Internet every second by 2019. However, even for now, video streaming users experience video stalling during video playback from time to time. According to Dobrian et al.[7], video stalling is the most dominant factor affecting the Quality of Experience (QoE), one of the major indices to evaluate network performance, of video streaming users. Therefore if we provide video packets a better forwarding route (e.g. a route with larger available bandwidth), the QoE of video streaming users may be improved. To achieve it, the controller needs to be able to identify video streaming packets in order to conduct a better forwarding strategy. Thus, discerning packets' content type is crucial for the purpose.

Stateful Packet Inspection (SPI) is a popular packet filtering technique. However, SPI only examines the header of a packet. HTTP traffics sent from the server sometimes do not contain the content type (MIME type) of a packet in the header. In this case, STI is not in a position to detect the packet's content type. There is another packet filtering method called Deep Packet Inspection (DPI) which examines both the header and the data part of a packet, and is frequently used to resolve the aforementioned problem STI encounters. DPI analyzes the data using the signature matching technique, and is commonly used in network security to block Trojans, viruses or malicious software. Jarschel et al. demonstrated how DPI can be used to identify video streaming traffic (specifically, YouTube videos) and improve the QoE of video steaming users.[3] However, DPI has its own disadvantages. For instance, in Jaschel et al.'s study, it takes about twenty seconds to identify video traffic and to inform the controller of the results. In addition to the prolonged inspection time, there are other limitations. For example, the speed of the computer may be reduced, because when a packet arrives, all network layers are parsed and inspected. This adds a burden on the processor and can seriously slow down the computer as this task requires many CPU cycles.[8] Besides, DPI is a complex technique that is hard to manage and maintain.

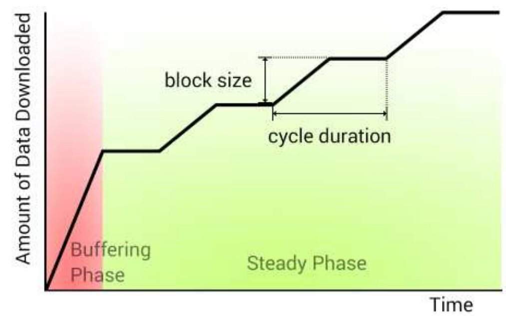

Fig. 2. Traffic pattern of video streaming service.

C. Traffic characteristics of Video Streaming Service

Internet video traffic is accountable for 64% of the total Internet traffic to date.[1] Therefore, the strategy used for video streaming can have a considerable impact on the Internet traffic. Currently, the most commonly used strategy for video streaming can abate the overhead on the network caused by the downloaded video that is unwatched and is believed to alleviate the load on the streaming server.[9]

The strategy produces a distinct traffic pattern as shown in Fig. 2[9]. There are two phases — the buffering phase and the steady phase. The buffering phase is commenced at the beginning of a video streaming session. During the phase, a burst of video data is received and it lasts for a few seconds. After the phase, the data receiving rate will be reduced and the session will enter the steady phase. During the steady state, video data is received periodically by blocks. The periodical receipt of data blocks produces ON-OFF cycles. In each cycle, video data is received in ON state and the connection is idle in OFF state. It is noticeable that the ON-OFF cycle duration and the data block size are fixed. Theoretically, average data received rate in steady state should be larger than the video encoding rate. Otherwise, the video playback will not sustain and the streaming user will experience video stalling.

III. System Scenario and Testbed setup

As illustrated in Fig. 3, there is an OpenFlow switch provides Internet connection to its user and handles traffic flows based on the flow table updated from the SDN controller. There are three different servers in our scenario

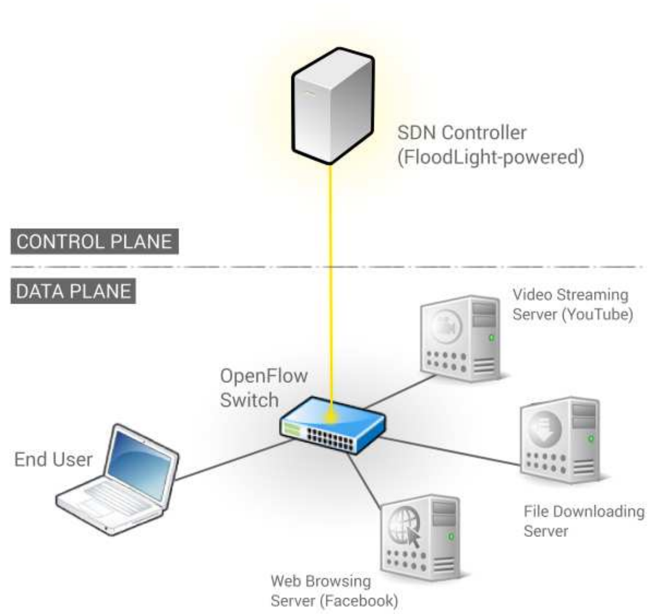

Fig. 3. The scenario of the experiment.

— the video streaming, file downloading and web browsing server. In the experiment, we choose the most popular video platform (YouTube) and social network website (Facebook) for the video streaming server and the web browsing server respectively.

The scenario of this study is designed that a user is watching a YouTube video, downloading large files and browsing Facebook at the same time. The manager of this network aims to identify the video streaming flows to optimize forwarding routes in the network. While performing these activities, the latency and success rate of video streaming produced by DPI and our strategy will be collected and compared.

IV. Traffic-Aware Networking for Video Streaming Service

The strategy used by most of the video streaming services produces an ON-OFF cycling traffic pattern with a fixed duration that is different to other activities(e.g., web browsing).[9] In the proposed technique, SDN statistics data are requested in every statistics retrieval time (denoted as *SRT*) which is a variable manipulated in this study. By monitoring the size of the data transmitted in *SRT* shown in the SDN statistics, a mechanism is designed which flags flows that appears to match the traffic pattern of video streaming services.

It is intuitive to think that lowering *SRT* as much as possible brings better results, because the traffic data collected will approach to a real situation. However, low *SRT* will increase the controller's frequency of requesting SDN statistics, and will therefore decelerate the controller and the switch. We register our concern about the trade off between the accuracy of matching traffic patterns and the load of the controller and the switch. However, how *SRT* affects the effectiveness of our mechanism is an empirical question, and this question was tested in this study.

As illustrated in Fig. 2, video streaming traffic retains a step-like pattern in the steady phase. By computing the growth rate of the data downloaded (Equation (1)), we notice a regular pulse pattern with fixed intervals. We proposed a mechanism using our technique to analyze the amount of data transmitted within *SRT* (denoted as DT_{SRT}) in every *SRT* to examine if the growth rate of DT_{SRT} (denoted as GR_{SRT}; defined at Equation (2)) exhibits the regular pulse pattern that is also shown in video streaming traffic.

$$GrowthRate = \frac{\partial^2(DataTransmitted)}{\partial(Time)^2} \quad (1)$$

$$GR_{SRT} = \frac{\delta DT_{SRT}}{SRT} \quad (2)$$

However, even if GR_{SRT} exhibits a regular pulse pattern, it cannot be guaranteed that the pattern of the flow's traffic appears to be the step-like pattern shown in video streaming traffic. For example, if the traffic of the flow appears to have a regular pulse pattern, GR_{SRT} computed will reveal the same pattern as well. To solve this problem, we compute the ratio of DT_{SRT} (denoted as R_{DT}) defined at Equation (3) instead. R_{DT} can reveal if the traffic of the flow has ON-OFF cycles.

$$R_{DT_n} = \begin{cases} \frac{DT_{SRT_n}}{DT_{SRT_{n-1}}} & n \in Z^+ \\ 1 & n = 0 \end{cases} \quad (3)$$

The system flow of the mechanism is illustrated in Fig. 4. At the first stage, the controller requests SDN statistics in every *SRT*. After retrieving the statistics data, the controller will run the rest of the stages for all the flows. R_{DT} of each flow is computed and recorded. The mechanism will then analyze if R_{DT} exhibits video streaming pattern. To subside false identification of flows, three parameters, confidence level (denoted as *CL*), the threshold for CL (denoted as TH_{CL}) and the decay rate (denoted as *DR*), are introduced to evaluate if a flow carries video streaming packets. *CL* is a parameter reflecting the similarity between the traffic pattern of the flow and typical video streaming pattern, and the value varies each time when running the mechanism. TH_{CL} is a chosen threshold value for *CL*. *DR* is a variable indicating the sensitivity of the mechanism. If the flow is determined to exhibits video streaming pattern, *CL* will increase, or in every *DR*, *CL* will decrease until it reaches zero. The flow will be flagged as a video streaming flow if *CL* passes its threshold TH_{CL}.

The pseudo code used for video streaming traffic identification is provided in Algorithm 1. *SRT* and *DR* are two variables specified when the algorithm starts. The flow is considered to exhibit video streaming pattern only if R_{DT} shows a pulse and is not sandwiched by zeros. This consideration is intended to forestall misidentified flows because of periodic pulses of data which happens in user activities such as browsing Facebook (which has spike-like patterns)[10] or website using Ajax to retrieve data from the server. If the flow is not considered to show video streaming pattern, *CL* will decrease. With this algorithm, flows having video streaming traffic can be promptly discerned.

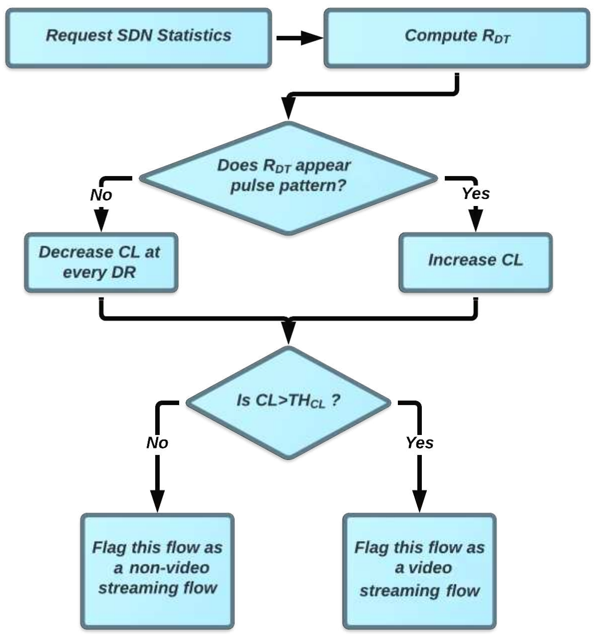

Fig. 4. The system flow of our mechanism.

Algorithm 1 Pseudo Code of Video Streaming Identification

> **function** PATTERNDETECT
>   **input** *SRT*, *DR*
> $CL \leftarrow 0$
> $TH_{CL} \leftarrow 1$
> *run*:
>   Retrieve SDN statistics.
>   **if** R_{DT} appears pulse pattern and is not sandwiched by zeros **then**
> $CL \leftarrow CL + 1$.
>   **if** $CL > TH_{CL}$ **then**
>   Mark the flow as a video streaming traffic.
>   **else if** $CL > 0$ AND R_{DT} does not appear pulse pattern in *DR* **then**
> $CL \leftarrow CL - 1$;
>   **goto** *run* in every *SRT*.

V. TESTBED SETUP AND EXPERIMENTAL RESULTS

A. Testbed Setup

The SDN testbed is built to evaluate the proposed traffic-aware networking technique. TP-Link WR1043ND SDN switch running OpenvSwitch 1.3 and Floodlight controller software[11] are used in the testbed. Floodlight is running upon Ubuntu 15.04 and the switch is connected to the controller by wire. The user receives and sends data to the switch via a wireless connection.

The user runs Safari 8.0 web browser for watching YouTube video and browsing Facebook. HTML5 is used for video playback and the videos played in our experiment are randomly chosen from YouTube. All the randomly chosen videos have playback time larger than one minute. The resolution option in the video player is configured as automatic mode. At the same time, a large file (over 100 megabytes) is downloaded and the user is browsing Facebook. The website is continually downloading text and picture files by Ajax request.

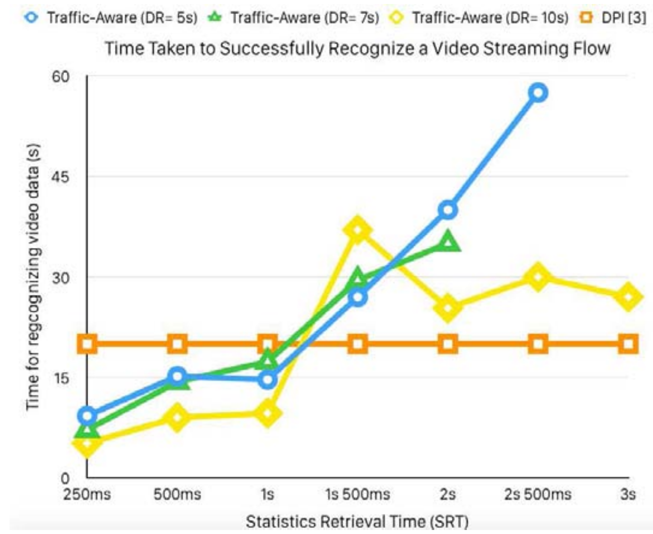

Fig. 5. The time taken to successfully recognize the video streaming flow.

B. Watching YouTube video

Fig. 5 shows the latencies for the proposed traffic-aware technique to recognize YouTube video streaming flows in various settings. The are three test groups each with different value of *DR*. For each group, there are seven cases each with a different value of *SRT*. Intuitively, *SRT* should be the dominant factor affecting the time taken to successfully recognize a video streaming flow, because with low SRT, the controller is able to collect more traffic statistics and is able to identify the flow faster. However, *DR* also affects the results. The case producing the best result occurs when *DR* is set as 5 seconds and *SRT* is set as 250 milliseconds. The case took 5.17 seconds in average to identify a video streaming flow, which is four times faster than the 20 seconds required by DPI.[3].

It is a significant improvement in the time required to identify video streaming flows. However, the amount of video data correctly identified as a video is also important for assessing the effectiveness of our proposed technique. Fig. 6 shows the percentage of video data transferred in one minute that is correctly identified (defined as the success rate). The result of the DPI case is conducted using the average time for DPI to identify video streaming data. We conducted experiments using the average identification time, which is 20 seconds, and found that 56.17% of the video data are correctly identified in one minute. All cases with *SRT* that is equal or less than one second outperform the case of DPI. When *DR* is set to 10 seconds and SRT is defined as 250 milliseconds, the success rate is 77.40%, which is 138% higher than the DPI counterpart.

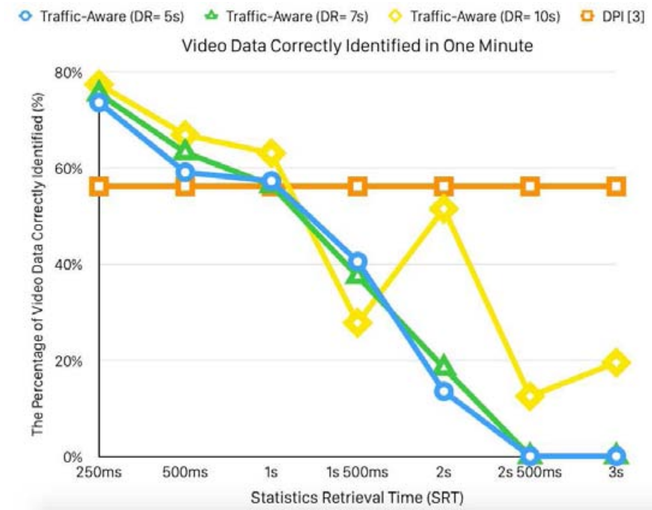

Fig. 6. Video data identified as video data in one minute.

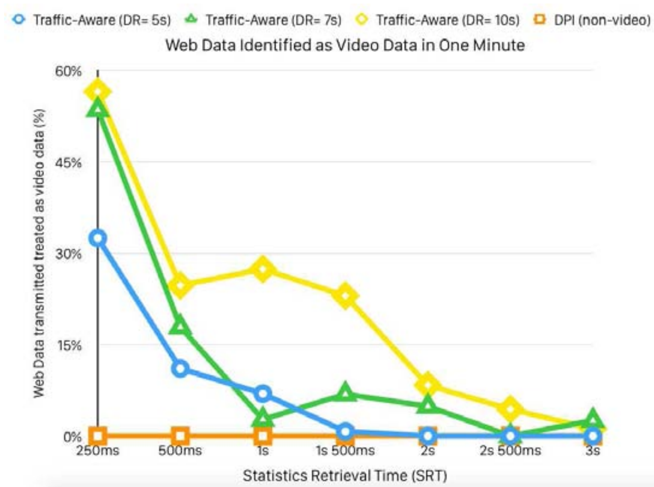

Fig. 7. Web data identified as video data in one minute.

C. Downloading large files

The experimental result showed that the flow of large file downloading will be 100% flagged as a non-video streaming flow for all the cases we tested. The result is on-par with the theoretical result of the DPI method.

D. Browsing Facebook

Fig. 7 shows the percentage of web data transferred in one minute that is identified as video data (defined as the false positive rate). Using the proposed strategy, cases with *SRT* set as 250 milliseconds show a high false positive rate that may be caused by a loosely chosen threshold. Thus, taken the false positive and the success rate, and the latency required to recognize video data into consideration, the case when *DR* is set as 7 seconds and *SRT* is set as 1 second exhibits the best result. Compared with DPI, this case yielded better success rate (56.37%) and shorter recognition latency (17.3 seconds), and a 2.7% false positive rate which is very close to the 0% theoretical value of DPI.

VI. Conclusion

In this study, a traffic-aware networking technique to promptly identify video streaming flow using SDN was proposed and tested. The experimental results show that with a certain *DR* and/or *SRT*, the proposed technique can reduce the latency to recognize video streaming flow and acquire a higher success rate. It is also noteworthy to mention that there exhibits a trade off between the success rate and the false positive rate. Nevertheless, the results of this study, i.e., latency, success rate and false positive rate, showed that compared with DPI, our technique is a strong competitor for identifying traffic data. Moreover, the proposed technique is also less complex compared with DPI and is therefore easier to extend and deploy. This work is a very first step in using video streaming as an example to examine the utility of the technique we proposed. In the future, machine learning and more user activities will be included in the study.

References

[1] "Cisco visual networking index: Forecast and methodology, 2014 âĂŞ 2019." [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html

[2] Z. Chen and X. Wang, "Effects of network capacity under variations of network structure and routing strategy," in *IEEE International Conference on Networking, Sensing and Control (ICNSC)*. IEEE, 2006, pp. 150–155.

[3] M. Jarschel, F. Wamser, T. Hohn, T. Zinner, and P. Tran-Gia, "Sdn-based application-aware networking on the example of youtube video streaming," in *Second European Workshop on Software Defined Networks (EWSDN)*. IEEE, 2013, pp. 87–92.

[4] H. Nam, K.-H. Kim, J. Y. Kim, and H. Schulzrinne, "Towards qoe-aware video streaming using sdn," in *Global Communications Conference (GLOBECOM)*. IEEE, 2014, pp. 1317–1322.

[5] "The history of java technology." [Online]. Available: http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html

[6] "Openflow, open network foundation." [Online]. Available: https://www.opennetworking.org/sdn-resources/openflow

[7] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 362–373, 2011.

[8] Y. H. Cho, S. Navab, and W. H. Mangione-Smith, "Specialized hardware for deep network packet filtering," in *Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream*. Springer, 2002, pp. 452–461.

[9] A. Rao, A. Legout, Y.-s. Lim, D. Towsley, C. Barakat, and W. Dabbous, "Network characteristics of video streaming traffic," in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*. ACM, 2011, p. 25.

[10] W. Wongyai and L. Charoenwatana, "Examining the network traffic of facebook homepage retrieval: An end user perspective," in *International Joint Conference on Computer Science and Software Engineering (JCSSE)*. IEEE, 2012, pp. 77–81.

[11] "Floodlight openflow controller." [Online]. Available: http://www.projectfloodlight.org/floodlight/