Janus - An MPEG-DASH Video Streaming Load Balancer based on Software-Defined Networking

Edenilson Jônatas dos Passos
Graduate Program in Apllied Computing (PPGCA)
Department of Computer Science (DCC)
Santa Catarina State University (UDESC)
Joinville, Brazil
edenilson.passos@yahoo.com

Adriano Fiorese
Graduate Program in Apllied Computing (PPGCA)
Department of Computer Science (DCC)
Santa Catarina State University (UDESC)
Joinville, Brazil
adriano.fiorese@udesc.br

***Abstract*—Recently, with popularisation of video streaming service, new video distribution technologies have been created. Currently, one of the most promising ones is the Moving Picture Expert Group Dynamic Adaptive Streaming over HTTP or MPEG-DASH. Even so, with the limitation of the TCP/IP network structure, the end user Quality of Experience (QoE) may be affected. One issue that can affect user QoE is the workload of content distribution servers. Thus, the unbalancing of server's workload comprising user's attendance can lead to a content server provider non optimised choice. This work presents two load-balancing solutions between MPEG-DASH video servers based on Software-Defined Networks, using as a balancing workload metric the throughput of the content server as well as the CPU load.**

***Index Terms*—Load balancing, MPEG-DASH, SDN, Flows manipulation**

I. Introduction

The use of the Internet on video transmission plays a relevant role in the business models of the current content providers. With the popularisation of streaming services, and consequently the overloading of the links as well as the restrictions imposed on this type of traffic by ISPs, companies began to develop adaptive video formats and content using the HTTP protocol as delivery method. Thus, the standard ISO/IEC 23009-1 or Moving Picture Expert Group Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [1] appeared.

A Content Distribution Network (CDN) is responsible for storing content that customers want as well as routing the requests to the closest server to the client. In this way, a CDN is responsible for load balancing between content servers [2], [3]. In these cases, load balancing is needed since, according to [4], server congestion is the factor that most affects the Quality of Experience (QoE) of the user because it is perceived as interruption and noise in the content's image and audio.

A contemporary approach to load balancing in CDNs for video content delivery is based on serving multiple client requests across multiple servers distributed in the CDN. To do this, the assignment of which server will attend to which client is done by means of the custom deployment of Domain Name Server (DNS) servers by the CDN provider.

Therefore, the problem of load balancing in CDNs, i.e., between content servers, in this particular case of MPEG-DASH video servers, suffers from management difficulties with the current traditional approaches. The contribution of this article aims to provide an architectural approach based on SDN for the solution of this problem. The balancing performed is intended to alleviate the workload of the content source servers and uses a combination of server throughput and CPU utilisation as balancing metric.

This paper is organised as follows. Section 2 discusses related work. Section 3 architecture and operation modes of the proposed solution are detailed. Section 4 presents the evaluation of the proposal. Finally, Section 5 presents final considerations comprising this work.

II. Related Works

In [5] an adapted Dijkstra algorithm is presented for load balancing on a CDN using SDN technology. With the proposed algorithm, the intention is to select the shortest path and with the least congestion.

Similar to [5], the central idea of [6] is to propose an algorithm in an SDN network that finds a more appropriate route to reach the best content distribution server.

Th use of artificial intelligence for load balancing is addressed in [7]. There, the SDN approach is used to redirect the video streams according to the result of the proposed load-balancing algorithm. That work's main goal is to save bandwidth by selecting the best possible path between client and a content server without congestion.

Load balancing based on metrics to choose the most appropriate server in general is developed in [8].

Thus, looking at the works surveyed on load balancing in content distribution systems (especially video), it is possible to note possibility of improvements. In particular, considering the SDN paradigm for the establishment of load balancing between content servers executed directly at the level of the network infrastructure (layers 2,3 and 4). Additionally, using the throughput and CPU utilisation metrics of the servers involved, as well as monitoring and using these values make

978-3-903176-23-2

load balancing highly dynamic and scalable, whether it is adding clients or servers.

III. Proposed Janus Solution

This section introduces the entitled Janus proposed approach based on SDN to load balancing between MPEG-DASH content servers.

A. Proposed Architecture: Static Mode

In static mode, the OpenFlow Ryu controller is the main actuator of the system. It performs the key functions such as obtaining the throughput metric of the involved video servers and deciding which one is the most appropriate to the client, generating and installing the traffic redirection rules in the OpenFlow switch.

Figure 1 presents the architecture of the proposed solution in static mode.In addition, Figure 1 indicates the numerical sequence of execution of these activities as a sequence of events.

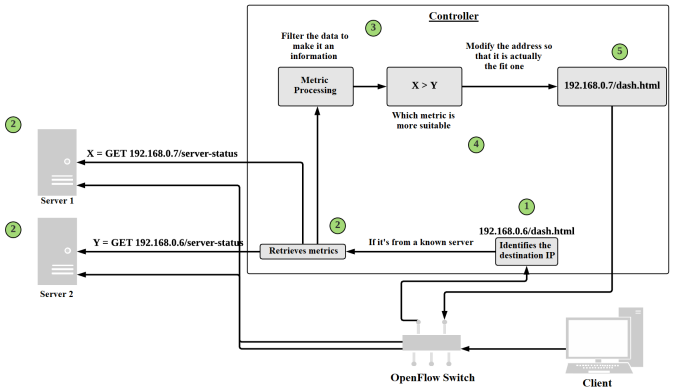

Fig. 1. Proposed Janus Load Balancing Architecture

In this way, according to the proposed architecture, the data packet representing the client request, upon arriving at the OpenFlow switch and forwarded to the controller, is checked in order to identify the requesting client and the content to be delivered. To do this, by establishing the TCP connection and executing the HTTP request of the content, a function identifies the source IP address (1). With the source address, the next step is to check the destination address (1). If the destination address is one of the known and available content servers, the next step is to obtain the load balancing metric values from these servers (2). Values of throughput of the servers is accomplished by the load balancing application, which is developed to the controller. This is performed a single time during the delivering of the video, soon after the process of obtaining the origin IP. That is, with each new TCP connection to the HTTP service, addressed to one of the servers requesting MPEG-DASH video, the metric values used are redeemed. Therefore, load balancing occurs only because of connection establishment (and consequent HTTP session).

After obtaining the values of the metric used, the balancing application evaluates which server has the highest throughput (3, 4) in order to choose one. This is due to the assumption that the one with the highest throughput is able to receive more connections and prone to lower latency. Furthermore, the next and final step is the generation and installation of the OpenFlow switch rules for IP and MAC addresses and the destination port of content traffic based on the best server chosen (5).

B. Proposed Architecture: Dynamic Mode

The dynamic redirection approach is relevant to load balancing since it allows greater flexibility in selecting the most suitable server to receive the MPEG-DASH connection. In this approach, the behaviour of the redirection process is similar to that of the static mode. That is, by noticing the possibility of redirecting to a server whose metric is most appropriate, the controller changes the address fields and logical ports in such a way that the data stream is transferred to and from the chosen server. However, in the process of dynamic redirection, besides the verification of the metrics of the servers when making a new connection, there is also a periodic monitoring of the metrics.

One of the limitations of the static approach is the inability to handle the transfer rate limit. For this reason, in the dynamic approach, the Equation 1 was idealised in order to attenuate such a limitation. It indicates that the metric used for load balancing (BC) is a composition between the throughput and the inverse of the server CPU utilisation. That is, the higher the throughput and the lower the CPU consumption the greater the chance that the server will be used to service a client.

$$BC_i = Ttransfer_i + \frac{1}{\%CPU_i} \quad (1)$$

The metric retrieving frequency was chosen based on the datasets used since they are 126 and 181 seconds long, respectively.

IV. Experiments and Results

A. Test Environment

All the experiments were performed on Linux virtual machines with help of the VirtualBox tool. Mininet emulator was used to simulate client hosts altogether with OpenFlow needs. Content servers were executed in a particular machine.

Regarding the available video/audio content, two datasets of Creative Commons (CC) license videos offered for testing by YouTube were chosen. The first dataset is called ***CAR CENC***. This video has 181 seconds of duration, and it is available in 6 video qualities (different resolutions) ranging from 256 $\times$ 144 to 1920 $\times$ 1080 pixels and an audio track. In this case, both video and audio are available in chunks of 2 seconds.

The second dataset is called ***FEELINGS VP9***. This dataset has VP9 video compression and therefore its size is considerably smaller than the previous one. However, it presents only high definition quality, i.e., Full HD is not available. This video content has 136 seconds of duration, and it is available in 6

video qualities ranging from 426 × 240 to 1280 × 720 pixels and an audio track. [1].

B. QoE Evaluation

In addition to the proposed approach of traffic-oriented load balancing, based on the throughput metric (static mode) and on the throughput and CPU utilization (dynamic mode), three other approaches were developed. One of them uses an algorithm to choose the server at random. There is also the Round Robin algorithm executed between MPEG-DASH content servers. Moreover, a third approach, called *Withouth SDN*, was also developed, making available an approach without load balancing.

To obtain the results, 10 tests were performed for each dataset of each approach and the mean value of those runs was used as a response. At each test, the controller was shut down just as the Mininet network and the browser cache in the clients was emptied.

The video bit rate variation was monitored comprising time. Bearing in mind the monitoring features implemented in the MPEG-DASH player, in this case, in particular the bit rate monitoring, for every execution a report with all the characteristics along the streaming is generated. After 10 executions for each dataset and for each approach, the calculation of the average of bit rate in each second of execution is performed. Therefore, in some situations in the graphs shown in Figures 2 and 3 the bit rate corresponding to different approaches, is superimposed.

However, when looking at Figure 2 it is noted that among the approaches tested, the one that obtained the lowest bit rate average is the one without SDN. The other approaches, however, were quite similar. At the beginning, from the second 1 to 10, all of them have similar behaviour due to the ladder effect. From the second 10 a slight variation between them is remarkable. Nonetheless, from the second 40 to the second 70, it is possible to notice a significant decrease in the bit rate average of the without SDN approach, as well as in the Janus dynamic one. The bit rate drops in the dynamic approach generally when server switching was performed, so in some situations due to this change the quality was decreased but soon returned to the highest one.

In the second dataset (parachute), according to Figure 3, the behaviour of all approaches were extremely similar. This is due to the fact that this dataset corresponds to a video whose highest resolution is 1280 × 720 pixels, which means it is smaller than the car dataset, in addition to the VP9 compression. For these reasons the download of this video was fast.

In general, according to the performed experiments, the proposed Janus architecture has shown to be promising since it can reduce the response time of the MPEG-DASH video content request, especially in the dynamic mode. Static mode can maintain the high bitrate, thus preserving the QoE. However, even with high performance it is the second method that consumes the most bandwidth, which is an important factor to take into consideration if it is implemented in production.

[1]In the results, when talking about dataset *car*, it refers to the dataset ***CAR CENC***. The ***FEELINGS VP9*** is read as *parachute*

V. Conclusion

With Internet popularization and technological advances related to architecture and network structure, video consumption has become a practice in the daily lives of many people. However, despite these advances and constant improvements, there are still several challenges to be overcome. One of them is load balancing so that the content distribution network can support a high demand for requests and does not detract from the end user Quality of Experience.

This work presents a possible solution or at least mitigation to this problem. To do so, an architecture that uses the Software-Defined Networking paradigm to intercept the packets during video content reproduction was devised, and using the analysis of the throughput metric and CPU processing rate of the available content servers, the most appropriate choice of server to deliver the content is possible.The main objective of the evaluation was to highlight that although it is a solution that modifies the connection flows, there is no perception of this redirection in the reproduction of the content consumed by the user and therefore of the Quality of Experience. In addition, the proposed architecture on its modes presents gains in several aspects such as response time, number of quality changes (video resolution) and video stalls.

Notwithstanding, more experiments are needed regarding the horizontal scalability of the proposed approach to verify its full applicability. Furthermore, it is intended to make comparisons with other approaches, as well as to increase the balancing decision by aggregating values of more criteria/metrics relevant to both the end-user QoE and video servers' better use of the computational resources.

References

[1] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, 04 2011.

[2] T. Bourke, *Server load balancing*, 1st ed. O'Reilly, 2001.

[3] V. Cardellini, M. Colajanni, and P. Yu, "Dynamic load balancing on web-server systems," *IEEE Internet Computing*, vol. 3, no. 3, pp. 28–39, 1999. [Online]. Available: http://ieeexplore.ieee.org/document/769420/

[4] B. Doshi, C. Kumar, P. Piyush, and M. Vutukuru, "Webq: A virtual queue for improving user experience during web server overload," in *2015 IEEE 23rd International Symposium on Quality of Service (IWQoS)*, June 2015, pp. 135–140.

[5] J.-R. Jiang, W. Yahya, and M. Ananta, "Load balancing and multicasting using the extended dijkstra's algorithm in software defined networking," *Frontiers in Artificial Intelligence and Applications*, vol. 274, pp. 2123–2132, 01 2015.

[6] E. Kaysudu, C. Çetinkaya, K. Hergüner, and M. Sayıt, "Server selection for video streaming applications over software defined networks," in *2016 24th Signal Processing and Communication Application Conference (SIU)*, 05 2016, pp. 1965–1968.

[7] C. Chen-xiao and X. Ya-bin, "Research on load balance method in SDN," *International Journal of Grid and Distributed Computing*, vol. 9, no. 1, pp. 25–36, 2016. [Online]. Available: http://www.sersc.org/journals/IJGDC/vol9_no1/3.pdf

[8] R. L. Carter and M. E. Crovella, "Server selection using dynamic path characterization in wide-area networks," in *Proceedings of INFOCOM '97*, vol. 3, April 1997, pp. 1014–1021 vol.3.

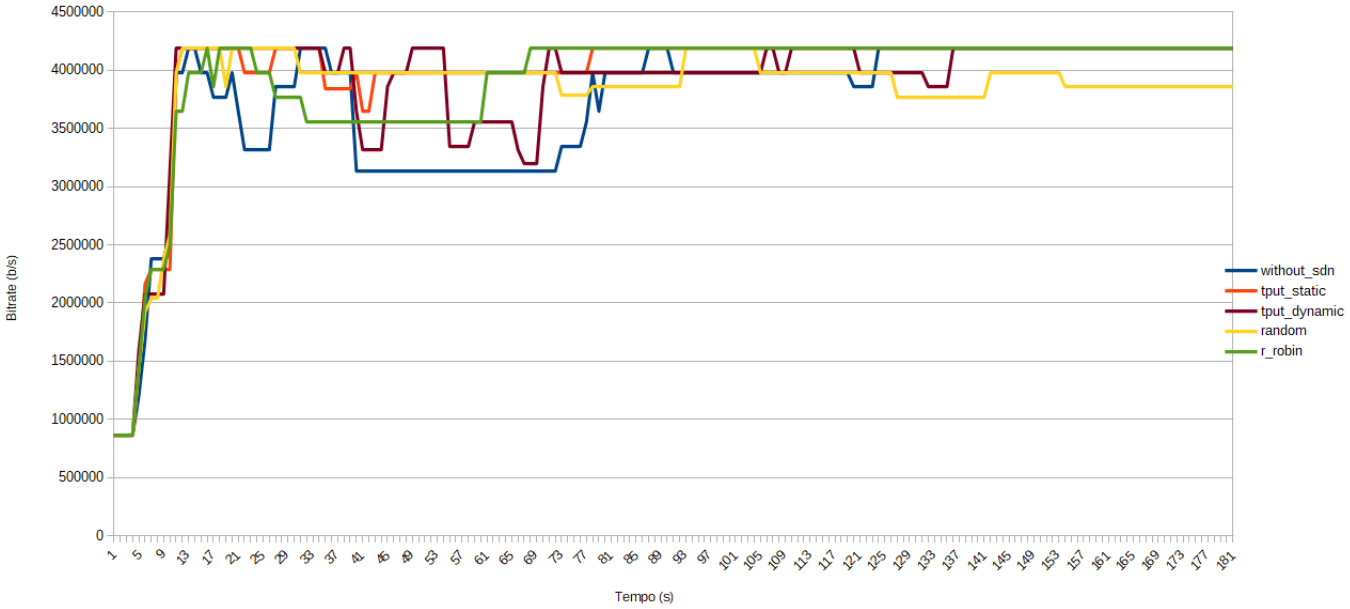

Fig. 2. Final result, average bitrate per approach - Car

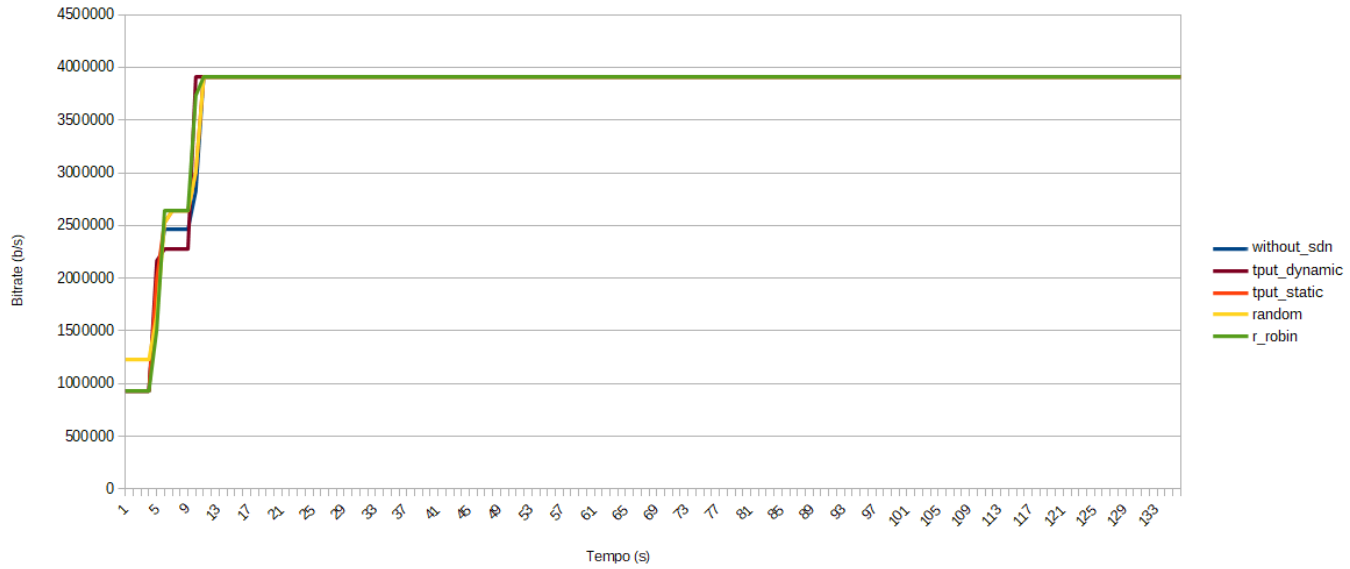

Fig. 3. Final result, average bitrate per approach - Parachute