Network-Dependent Server Performance Analysis of HTTP Adaptive Streaming

Conference Paper · September 2017

DOI: 10.1007/978-3-319-67597-8_7

Saso Gramatikov
Ss. Cyril and Methodius University in Skopje
32 PUBLICATIONS **60** CITATIONS
SEE PROFILE

Network-dependent Server Performance Analysis of HTTP Adaptive Streaming

Sasho Gramatikov

Faculty of Computer Science and Engineering,
`sasho.gramatikov@finki.ukim.mk`

Abstract. The HTTP adaptive streaming (HAS) is a popular mechanism for delivery of live and on-demand video contents encoded with different qualities and divided into segments with equal length. The mechanism adapts the requested segment qualities to the quality of the link, providing uninterrupted service even in congested network conditions. In this work, we analyze the HAS for delivery of Video on Demand (VoD) contents from server performance point of view for different segment lengths and different network conditions. For that purpose, we created an environment for real-case measurements of the server performance and measured performance parameters like CPU utilization, generated in-bound and out-bound traffic and number of established TCP connections. From the analysis of the obtained data, we conclude that streaming of shorter video segments generates more appropriate and predictable traffic pattern, but requires more CPU power and TCP connections. Therefore, the shorter contents are suitable for streaming in networks with very low packet losses. Longer video segments, on the other hand, tend to require more resources only at the beginning of the streaming session, which they release before the end of the session, and hence, alleviate the network equipment. The main advantage of using long segments is that they can achieve uninterrupted streaming experience even in harsh network environments such as congested wireless networks.

Keywords: Adaptive streaming, Video-on-Demand, Performance

1 Introduction

The HTTP adaptive streaming (HAS) emerges as a response to the large popularity of the video streaming services and the necessity to provide uninterrupted video experience in networks with different bandwidth capacities for devices with different capabilities. It is a hybrid mechanism that takes the advantages of the traditional and the progressive streaming.

In the HAS, each video is encoded with several different qualities, and each quality is divided into segments with the same length, but different size. The segments typically have duration between 2 and 10 seconds. Apart from division of the videos, a manifest file containing relevant information about the available qualities of the video and the URL of the segments is created. The clients request the manifest file, and based on its content, their video player proceeds requesting segments with size appropriate to the network conditions. Hence, the devices with high resolution displays and high-speed Internet connection require the best quality segments, while the mobile devices with limited bandwidth require the lowest quality segments. The main feature of the HAS that makes it superior to the other streaming mechanisms is that the devices can switch to lower qualities whenever the player detects that the network conditions deteriorate and switch back to the maximal quality when the network recovers. Although reducing the quality of the video reduces the Quality of Experience (QoE), it keeps the clients satisfied because it still provides uninterrupted service. Just like the progressive streaming, the HAS uses the HTTP as application protocol to deliver the video segments, i.e., the segments are equally treated as web pages. Therefore, the videos can be hosted on ordinary web servers for hosting web contents, instead of use of proprietary streaming servers typical for the traditional streaming.

Another huge advantage of the adaptive streaming comes from the fact that the HTTP uses TCP as a transport protocol, which guarantees that the data between the server an the client will arrive in order and with no errors. Moreover, since the TCP uses the well known port number 80, the video contents can easily pass through the firewalls of even the most restrictive administrators.

Unlike the progressive streaming, which tends to maximally utilize the available throughput enabling the clients to download significantly higher portion of the video at the very beginning of the transfer, the HAS tends to entirely download only one segment at a time with the maximal available speed. Therefore, whenever the clients desire to stop the video session, no downloaded data is wasted as it is the case with the progressive streaming.

The only disadvantage of the HAS streaming is that it requires extra storage space for hosting many qualities of the some video. However, taking into account the cost of storage per byte nowadays, this disadvantage is not a great concern for the video service providers.

There are many commercial HAS implementations such as Apple's HTTP Live Streaming (HLS), Microsoft's HTTP Smooth Streaming (HSS) and Adobe's HTTP Dynamic Streaming (HDS) used in the popular video streaming platforms. In order to overcome the different segment and manifest file formats, the HAS was standardized by MPEG into the open standard MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [10]. The standard defines the format of the manifest file called Media Presentation Description (MPD). The DASH standard is currently widely accepted by a large community of leading streaming companies which formed the DASH Industry Forum (DASH-IF) [1]. The DASH-IF provides open-source software tools for implementation of the DASH, which is the reason why we use DASH as a representative of the HAS mechanisms in our work.

The main interest of our work is to find out how the streaming server performs when streaming videos with different segment lengths under different network conditions. There is a vast community that treats the HAS streaming with main focus on performance of various algorithms used by the players to choose the appropriate video qualities. The goal of these algorithms is, on one hand, to provide uninterrupted service even in unfavourable network conditions, and on the other hand, to reduce the number of quality switches. Thus, in [8][6][4][11] the authors analyze the QoE for DASH streaming of a single video session for various bandwidth restrictions, packet delay and packet loss probability. In [9][12][5], the authors propose various algorithms for choosing the optimal quality levels in order to achieve the aforementioned goals, comparing not only a single session, but also competitive requests that share the same link by varying the network conditions and other algorithm-specific parameters. However, to our knowledge, there are no articles in the literature that treat the measurement of server performance for simultaneous service of large number of clients. Therefore, the goal of this paper is to measure and compare the resources required by the server for VoD streaming using HAS, emphasizing the effect of the network conditions and the video segment duration. For that purpose, we created test environment consisting of a streaming server and clients that simultaneously generate requests for videos from the server. Then, we measure the resources required by the streaming server such as CPU utilization, out-bound, in-bound bandwidth and number of established TCP connections in different emulated network conditions. The results of these measurements give an overall picture of the efficiency of the HAS, which can serve the VoD service providers in profiling their streaming servers under different network conditions.

The rest of the paper is organized as follows: in Section 2 we describe the network environment, the software and the content adaptation required for conducting the measurements. Afterwards, in Section 3, we analyze the obtained results and give the conclusions from our work in Section 4.

2 Measurement Environment

In order to create real-case scenarios for measurements of adaptive streaming, we designed a simple network consisting of a web server that hosts videos with different segment duration and clients that generate requests for the videos. The server is a virtual machine with 4 CPU cores and 16 GB or RAM. Since the adaptive streaming does not require dedicated video streaming server, we use the Apache web server that runs on Ubuntu 16.04 operating system.

We use 3 client machines that are connected to the server on the faculty campus network via 4 hops. The overall capacity of the links between the clients and the server is limited to the speed of Fast Ethernet used to connect the clients to the network. The small number of clients generate frequent requests in a short time interval to simulate the streaming process of large number of clients that request a single video . There are 42 clients that generate requests for videos with a rate of 1 request per second.

In order to measure the server performance in different network conditions, we use the built-in Linux Kernel module *netem* [7] that runs on the streaming server. It emulates different network conditions by adding configurable delay or packet loss probability to every packet that passes through the network interface.

We measure and record the system performance during the streaming process by using a dedicated program that runs on the server. The program uses the SIGAR library [3] which provides a cross-platform, cross-language programming interface to low-level information on computer hardware and operating system activity. The program periodically gathers the performance data and writes it in a text file that is used for latter analysis.

The clients use the Mozilla Firefox web browser to request videos and its native video player to play the received data. The main reason for choosing Mozilla Firefox is that it is among the few browsers that allow establishing a configurable number of simultaneous TCP connection to the same host. This feature is essential for conducting the experiment where only a few clients establish large number of simultaneous connections to the same server.

Since the HAS streaming is not explicitly supported by the native players of the web browsers, we use an open-source JavaScript library for DASH streaming, developed and maintained by the DASH Industry Forum [1]. The library processes the requested manifest file and uses its contents to request video segments which are further played by the native video player.

The server hosts one sample 720$\times$680 video with duration of 60 seconds. The video is encoded with the H264 codec in MP4 format with playback rate of 1.1 Mbps and total size of 8.2 MB. We used the open-source MP4Box tool [2] to divide the video in segments with duration of 2, 5, 10, 30 and 60 seconds, and to create a MPD manifest file for each duration of the segments. We consider only a single quality video, i.e., there are no rate switches when the network conditions change.

3 Results and analysis

For each simulation scenario, we conduct three independent measurements and use their mean value in the analysis. In order to reduce the periodic spikes that appear in most of the analyzed parameters, we apply a smoothing function that calculates a moving average of the proximate measurement data.

In our first scenario, we measure the out-bound traffic originating from the streaming server for delivery of videos with different segment length. The results from the measurements are shown in Figure 1. The figure shows that as the segment length increases, so does the peak value of the traffic requested by the server. This peak is reached earlier in the time for longer segments. On the contrary, for segments with length of 2 seconds, the amount of traffic increases slowly and reaches the peak in the interval when all the clients in the system play the video. The peak value in this moment equals the sum of the playback rates of all clients in the system. The adaptive streaming with small segment length has the properties of traditional streaming, which downloads the video data with rate equal to its playback rate.

In the case when the segment length equals the length of the video, the peak is reached shortly after the first initiated requests as a result of the tendency of the TCP to maximally utilize the available bandwidth between the two end-

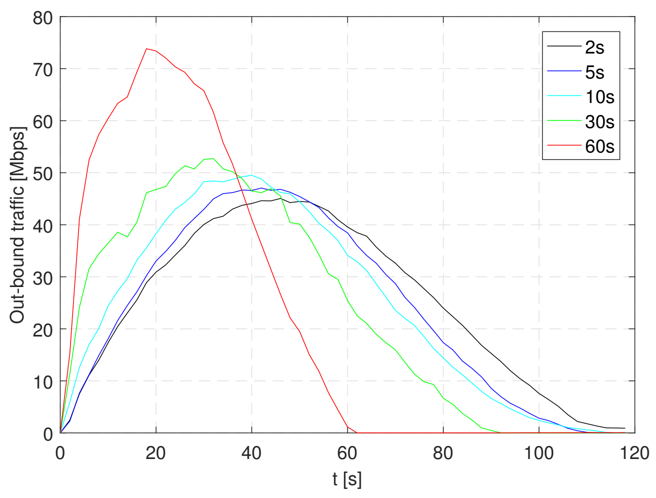


Fig. 1. Out-bound traffic for different segment duration

points. Thus, every client buffers the entire video at the beginning of the session and generates no traffic until the end of the session. Therefore, the curve for out-bound traffic reaches value zero shortly after the last request for video. In the moments when all the clients are simultaneously playing the video, there is no load on the server. The system has similar behaviour when the clients request videos with segment length of 30 seconds, but the out-bound traffic reaches the peak at the moment when part of the clients that reached half of the video request the second half, and the clients that initiate the session in that moment request the first half.

From the results of Figure 1, we can conclude that having short segments does not overload the network, requiring maximum bandwidth that equals the total playback rate of the videos. In this situation the traffic demands for a certain number of simultaneous session can be easily predicted, unlike the long segments which can lead to initial delays due to the tendency to use the available bandwidth. Another advantage of short segments is that if the clients decide to end the session at the middle of the streaming, there is no waste of transferred data since the video segments are downloaded as they are watched. In the case of long segments, the link is pushed to its capacity limits. However, the clients initially obtain large amounts of the video, which gives them enough time to load the rest of the video in case that the link saturates and, hence, the download rate becomes lower than the playback rate. Other disadvantage of long segments is that there is waste of the downloaded data in case that the client decides to end the session.

Another aspect of the traffic in the network is the number of established TCP connections for delivery of the video segments with different lengths shown in Figure 2. Although the short segments proved to generate lowest amount of peak out-bound traffic, they dominate with the number of established concurrent TCP connections. As the playback of the video proceeds, the player establishes a new connection for every subsequent segment. Upon reception of the segment, the player releases the connection. Hence, when the segment duration is very short, the connection establishment frequency increases, which leads to high number of concurrent connections. As expected, the peak number is reached when all the clients are receiving a video, which is at the middle of the first request and the end of the last session. Although there are 42 simultaneous streaming sessions at the time of the peak value, there are only approximately 9 connections since at the time of the measurements only that number of clients are downloading segments while the rest of the active clients are playing the buffered segments downloaded previously. The same analogy applies to the low number of connections for long video segments. In this case there are fewer segments that have to be downloaded, which imply lower number of established connections. Therefore, in Figure 2, the segments with duration of 60 seconds reach a peak number of approximately 4 connections. Each connection in this case is used to load entire segment and is never used by the client later during the session.

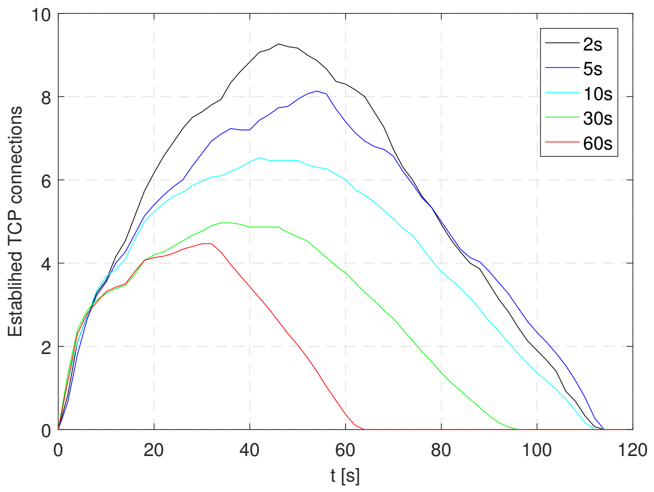


Fig. 2. Number of established TCP connection for different segment duration

The CPU utilization is another performance parameter considered in our work. For every video, the web server initially receives request for the MPD manifest file, and based on its content, continues receiving requests for the subsequent video segments. The number of requests depends on the length of the video, and therefore, it is expected that the web server will utilize different processing power to handle the requests. The CPU utilization for serving videos with different segment lengths is shown in Figure 3. The CPU is loaded the most when serving videos with segment length of 2 seconds. It reaches the peak value before the last request for video and slowly decreases to zero utilization at the end of the last session. The curves for CPU utilization for longer segments have similar shape in the initial part of the measurements since all sessions start with request for the MPD manifest file. The maximum difference in their peak values is approximately 1.5%. What makes the curves different is the descending shape after the peak value is reached. The videos with segment length of 60 seconds rapidly release the CPU before the sessions end, while the shorter segments start releasing the CPU with higher intensity, and then, slowly decrease to value zero. It is interesting to note that the peak of the CPU utilization does not temporally coincide with the out-bound traffic and the number of TCP connections peaks. From the figure, we can conclude that longer segments are more appropriate for use than shorter segments from CPU utilization point of view because not only they reach lower peak values than the short segments, but they also keep the server's CPU busy shorter time.

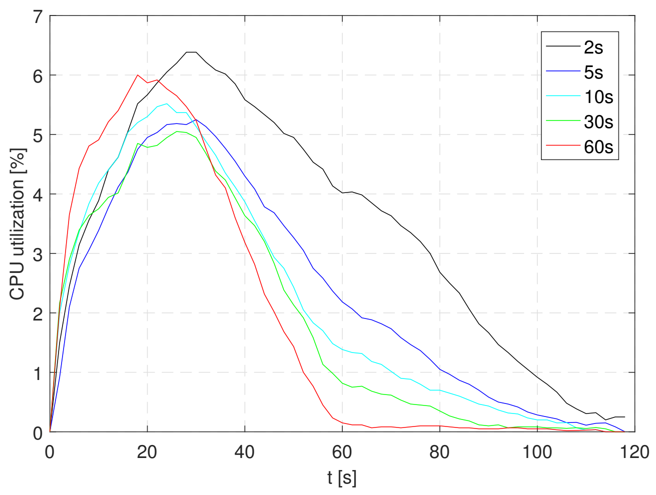


Fig. 3. CPU utilization for different segment duration in perfect network conditions

In order to analyze the server performance in harsh network conditions, we conducted the same measurements, but we introduced emulated packet loss with probabilities of 2, 5, 10, 12 and 15%.

In Figure 4, we show the out-bound traffic behaviour in unfavorable network conditions with packet loss probability of 10% for streaming videos with different segment lengths. The figure shows that the segments with lengths of 2 seconds are not appropriate for use in networks with high packet loss, which can be proved by the fact that there is still significant amount of out-bound traffic after the expected finish time of all sessions. This traffic implies that the clients experience stalls during the session. The reason for the stalls is that within the short duration of each segment, there is not enough time for the lost packets to be re-transmitted. Apart from the increased stall time, we can observe that the peak values of the traffic are reduced compared to the perfect network conditions as shown in 1, since the average time for delivery of one segment is increased, and hence, the average download rate per session is decreased.

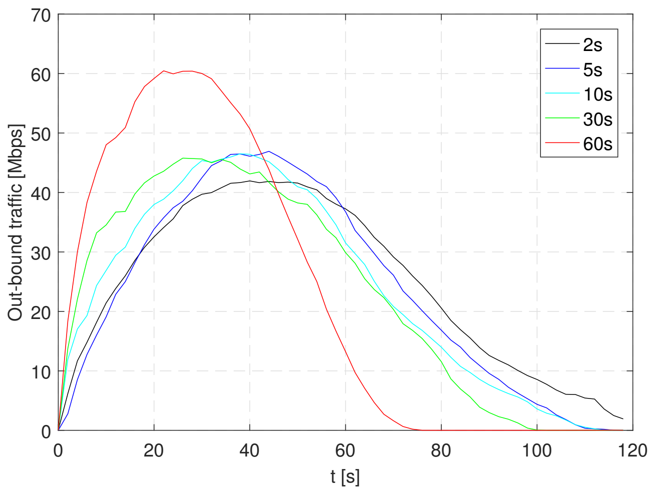


Fig. 4. Out-bound traffic during time for different segment duration with packet loss probability of 10%

The dependence of the out-bound and in-bound traffic on the packet loss probability for segments with duration of 2 seconds is shown in Figure 5(a) and 5(b), accordingly. In Figure 5(a), we can see that the short segments can be delivered to the clients in time only in perfect network conditions since, as the packet probability increases, the peak out-bound traffic reduces. This fact implies delay in completing the whole segments, and hence, interruptions in the playback of the video. Another indicator of the increased number of session interruptions is the presence of traffic after the predicted end of the last session.

The packet loss probability has also a significant impact on the server in-bound traffic as shown in Figure 5(b). The reason for the increased amount of received traffic for higher loss probabilities is that the clients send more requests for re-transmission of the unacknowledged TCP segments.

Unlike the short segments, the longer segments are more resistant to packet loss, which can be seen from the traffic patterns for segments with duration of 60 seconds in Figures 6(a) and 6(b). We can witness that the streaming process can withstand packet loss probabilities as high as 10% without interruptions of the watching experience. The only drawback of the long segments in harsh network conditions is that the clients may suffer a short initial delay until there is enough buffered data to play the video, however, this issue is mostly tolerable as there are

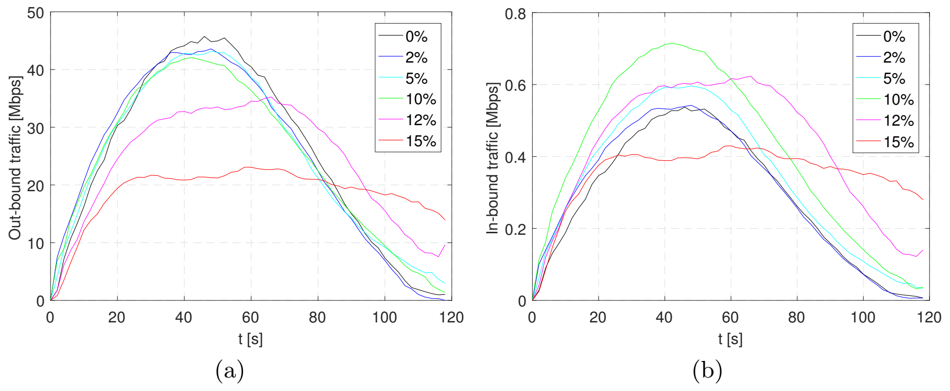


Fig. 5. (a) Out-bound and (b) in-bound traffic for different packet loss probabilities for segments with duration of 2 seconds

no further interruptions. The figure also shows that for packet loss probabilities above 12% the clients will suffer intolerable number of interruptions.

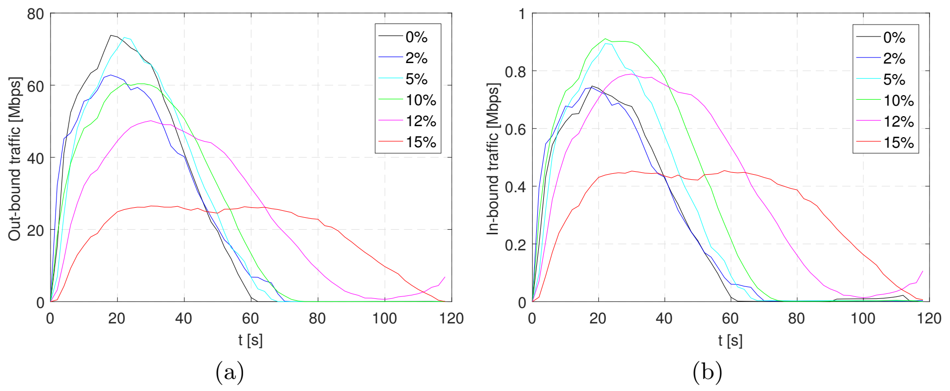


Fig. 6. (a) Out-bound and (b) in-bound traffic for different packet loss probabilities for segments with duration of 60 seconds

In Figure 7(a) and Figure 7(b), we show the impact of the packet loss probability on the number of established concurrent TCP connections for segment duration of 2 and 60 seconds. We can see that high packet loss probability implies higher number of TCP connections compared to the low packet loss probability. The reason for this behaviour is the fact that each client holds the connection longer time as a result of re-transmission of the unacknowledged segments. The connection is released upon a complete video segment delivery, and therefore,

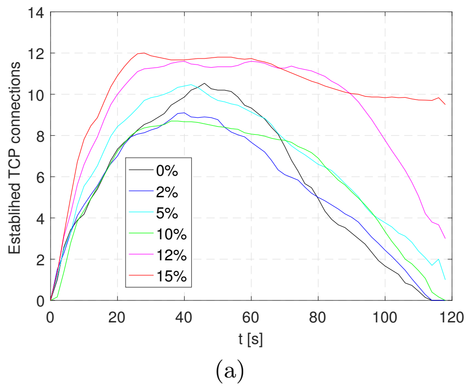
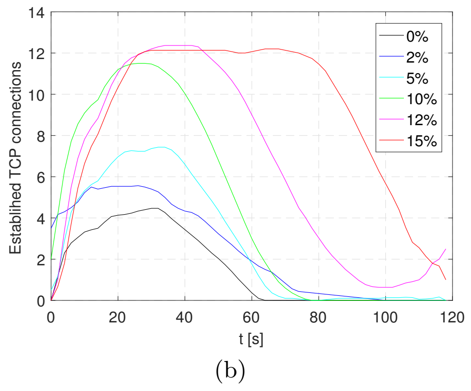
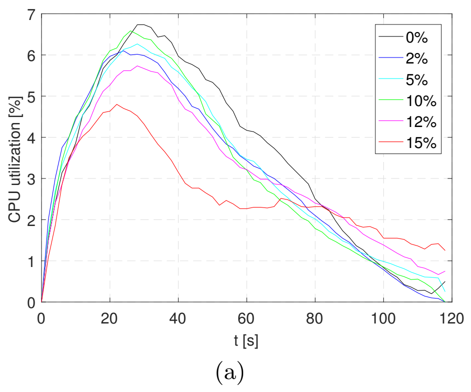


Fig. 7. Number of established TCP connection for different packet loss probabilities for segments with duration of (a) 2 and (b) 60 seconds



the more packets are lost, the longer it will take the server to deliver the entire segment.

Figures 8(a) and 8(b) show the impact of the packet loss probability on the server CPU utilization for serving the videos for segment duration of 2 and 60 seconds. The general conclusion from the figures is that, in cases of better network conditions, the CPU is kept longer time busy serving the requests that arrive with appropriate timing, unlike the cases with harsh conditions where some of the requests are delayed, and so is their processing by the CPU.



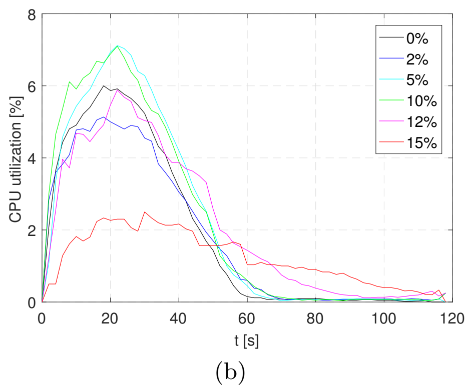


Fig. 8. Server CPU utilization for different packet loss probabilities for segments with duration of (a) 2 and (b) 60 seconds

Increasing the packet loss probability implicates more interruptions of the streaming sessions in general, however, the results obtained from our measurements show that certain segment lengths are more tolerant to packet loss than others. Table 1 shows an overview of the dependence of the service continuity for different segment lengths on the packet loss probability. From the results we can conclude that short video segments are only suitable for delivery of videos in reliable networks. As the segment length increases, so does the resilience to the packet loss. According to Table 1, the optimal segment length for unreliable networks is 10 seconds. Longer video segments also show satisfactory results, but their main downside is that the clients my suffer long initial delays. From the results we conclude that the HAS streaming with segment length of 10 seconds can be used to provide uninterrupted streaming service in congested wireless networks.

Table 1. Overview of segment lengths eligibility for uninterrupted streaming for different packet loss probabilities

	packet loss probability					
seg. duration	0%	2%	5%	10%	12%	15%
2s	yes	no	no	no	no	no
5s	yes	yes	yes	yes	no	no
10s	yes	yes	yes	yes	yes	no
30s	yes	yes	yes	yes	no	no
60s	yes	yes	yes	yes	no	no

4 Conclusion

In our work, we analyzed the influence of the video segment duration and the network conditions on the server performance for delivery of VoD contents using the HTTP adaptive streaming mechanism. For that purpose we created an environment for measurement of the server performance for serving clients in the campus network under different emulated network conditions.

From the analysis we concluded that hosting videos divided in shorter segments is more appropriate in better network conditions because they do not generate spikes in the required server traffic, i.e., the server traffic requirements are even and easily predictable. Their main disadvantage of short segments is that they require more CPU power and more resources for establishing TCP connections. Longer videos, on the other side, generate traffic patterns that can congest the network, but require less CPU power and fewer TCP connection. Their strong point is their resilience to unfavourable network conditions, which can be of great importance in congested wireless networks. The trade-off for providing predictable traffic patterns, fewer TCP connections, modest CPU utilization, and what is most important, uninterrupted watching experience is dividing the videos in segments with duration of 10 seconds. The results of our work can be a useful guideline for the VoD service providers in planing the server requirements for predefined demands and network conditions.

Acknowledgement

The author thanks the Faculty of Computer Science and Engineering at the Ss. Cyril and Methodius University in Skopje, under the EEAVS ("Energy Efficiency of Adaptive Video Streaming") project for financial support.

References

1. DASH Industry Forum (2016), `http://www.dashif.org`
2. MP4Box (2017), `https://gpac.wp.imt.fr/mp4box/`
3. SIGAR (2017), `http://sigar.hyperic.com/`
4. Biernacki, A., Tutschku, K.: Performance of HTTP video streaming under different network conditions. Multimedia Tools and Applications 72(2), 1143–1166 (2014)
5. Garcia, S., Cabrera, J., Garcia, N.: Quality-Control Algorithm for Adaptive Streaming Services Over Wireless Channels. IEEE Journal on Selected Topics in Signal Processing 9(1), 50–59 (2015)
6. Juluri, P., Tamarapalli, V., Medhi, D.: Measurement of Quality of Experience of Video-on-Demand Services: A Survey. IEEE Communications Surveys & Tutorials 18(c), 682–686 (2015)
7. Jurgelionis, A., Laulajainen, J.P., Hirvonen, M., Wang, A.I.: An empirical study of NetEm network emulation functionalities. Proceedings - International Conference on Computer Communications and Networks, ICCCN (2011)
8. Kesavan, S., Jayakumar, J.: Dynamic Adaptive Streaming Over HTTP (DASH)-Comprehensive Study and Rate Adaptation Performance Analysis. International Journal of Soft Computing 10(4), 261–273 (2015)
9. Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A.C., Oran, D.: Probe and adapt: Rate adaptation for HTTP video streaming at scale. IEEE Journal on Selected Areas in Communications 32(4), 719–733 (2014)
10. Vetro, A., Sodagar, I.: Industry and Standards The MPEG-DASH Standard for Multimedia Streaming Over the Internet. IEEE MultiMedia 18(4), 62–67 (2011)
11. Yitong, L., Yun, S., Yinian, M., Jing, L., Qi, L., Dacheng, Y.: A study on Quality of Experience for adaptive streaming service. Communications Workshops (ICC), 2013 IEEE International Conference on pp. 682–686 (2013)
12. Zhou, C., Lin, C.W., Guo, Z.: MDASH: A Markov Decision-Based Rate Adaptation Approach for Dynamic HTTP Streaming. IEEE Transactions on Multimedia 18(4), 738–751 (2016)