Evaluation of Gateway-Based Shaping Methods for HTTP Adaptive Streaming

Chiheb Ben Ameur, Emmanuel Mory, Bernard Cousin

Evaluation of Gateway-Based Shaping Methods for HTTP Adaptive Streaming

Chiheb Ben Ameur
Orange Labs
Rennes, France
chiheb.benameur@orange.com

Emmanuel Mory
Orange Labs
Rennes, France
emmanuel.mory@orange.com

Bernard Cousin
IRISA, University of Rennes 1
Rennes, France
Bernard.Cousin@irisa.fr

Abstract— **HTTP Adaptive Streaming (HAS) is a streaming video technique commonly employed over best-effort networks. However, it is characterized by some issues that harm its quality of experience (QoE) in cases of daily use. The main use case of the present investigation involves HAS clients competing for bandwidth inside the same home network. Based on related work, we found that one of the most convenient solutions for this use case is to define a bandwidth manager, on the gateway side, that divides the available home bandwidth between HAS clients. Two main methods have previously been proposed to shape the HAS streams in accordance with the bandwidth manager's direction and are referred to as gateway-based shaping methods: a highly renowned method, Hierarchical Token Bucket Method (HTBM), that uses the hierarchical token bucket queuing discipline, and another method, Receive Window Tuning Method (RWTM), that employs TCP flow control by handling only acknowledgment TCP packets. In this paper, we compare these two shaping methods. Results indicate that RWTM improves the QoE better than HTBM and does not add queuing delay. Results were validated through experimentation and objective QoE analytical criteria.**

Keywords— Traffic Shaping; Quality of Experience; HTTP Adaptive Streaming; TCP Flow Control; Bandwidth Management

I. INTRODUCTION

HTTP Adaptive Streaming (HAS) is a streaming video technique based on downloading video segments of short playback duration, called chunks, from a HAS server to a HAS client. Each chunk is encoded at multiple encoding bitrates, also called video quality levels. After requesting a chunk using an HTTP GET request message [1], the player on the client side stores the chunk into a playback buffer. The player operates in one of two states: Buffering State and Steady State. During the Buffering-State, the player requests a set of chunks consecutively until the playback buffer has been filled. However, during the Steady State, the player requests the chunks periodically to maintain a constant playback buffer size. This periodicity creates periods of activity, called ON periods [2], [3], followed by periods of inactivity, called OFF periods [2], [3], without impacting the continuity of the video playing. The player selects the quality level for each chunk by estimating the available bandwidth during the previous ON period.

The Quality of Experience (QoE) of HAS users can be evaluated by three main criteria:

1- Video quality level stability [2], [6]: A frequent change of video quality level bothers the user. Therefore, quality level fluctuation should be avoided to improve the QoE.

2- Fidelity to optimal quality level selection: The user prefers to watch the best quality level of video available. Accordingly, the HAS player should select the optimal quality level that has the highest feasible quality level permitted by the available bandwidth.

3- Convergence speed [6]: The user prefers to achieve watching the optimal quality level as soon as possible. Therefore, the HAS player should efficiently select this level. The player's delay to attain the optimal quality level is called the convergence speed [6].

Video packets sent from the HAS server to the HAS client pass through many network devices. Each device has one or many queues, as well as an algorithm called a "queueing discipline" that enables the scheduling of packets into the queue. The queuing discipline also decides whether to buffer, route or drop incoming packets to better manage the queue and to avoid network congestion. Bottleneck points are most likely to be located in the Digital Subscriber Line Access Multiplexer (DSLAM) [12]. In fact, DSLAM may considerably reduce its allocated bandwidth when it must divide it between many subscribers (DSL routers). Accordingly, DSLAM is more likely than other network devices to drop packets when the backbone network is well-designed. To reduce network congestions, the TCP protocol implements in the sender machine a congestion control protocol that reduces the sending rate when a congestion event is detected. However, this reduction of bitrate may degrade QoE. Moreover, the ON-OFF periods that characterize HAS players during the Steady-State phase involve false estimations of the available bandwidth. In fact, during the OFF periods, the player cannot estimate the available bandwidth, which may change over time. Furthermore, when HAS clients are connected to the same DSL router, the bandwidth estimation becomes more difficult, especially when the ON period of one HAS client coincides with the OFF period of another HAS client, which leads to bandwidth overestimation, congestions, and consequently, the degradation of the QoE. As explained in [2], the competition between HAS clients in a home network causes an instability of quality level selection, unfairness between players, and bandwidth underutilization.

The objective of the present study is to improve the user's experience (QoE) when HAS clients compete for available home bandwidth. Our methodology involves comparing between proposed methods, mainly gateway-based methods, by using three objective metrics of QoE in accordance with the description provided above. The remainder of this paper is organized as follows. In Section II, we describe and critique recent works addressing related methods. In Section III, we define a well-optimized bandwidth manager and describe the gateway-based shaping method Receive Window Tuning Method (RWTM) [13]. Section IV presents the experimental implementation that was used. Section V provides a detailed evaluation of results. In Section VI, we conclude the paper and suggest future directions to extend this work.

II. Related work

Many research studies have been conducted to improve the QoE when several HAS clients are located in the same home network. The methodology most often employed involved avoiding false estimations of the available bandwidth during ON periods in order to more effectively select the video quality level. Three types of solutions were proposed to improve HAS user QoE: client-based, server-based and gateway-based solutions. These differ with respect to the device in which the solution is implemented. Below, we cite relevant methods for each type of solution:

- One of the recent client-based solutions is proposed in the FESTIVE method [4]. It randomizes the events of chunk requests inside the player in order to reduce the periodicity of ON periods. Consequently, most of the incorrect estimations of bandwidth could be avoided when several HAS clients competed for bandwidth. However, this method is not efficient enough to prevent all incorrect estimations. Moreover, FESTIVE does not provide coordination between HAS clients, which is required to further improve bandwidth estimations and QoE.
- The authors of [5] propose a server-based method: it consists of detecting the oscillation between quality levels on the server side and deciding which optimal quality level must be selected. Although this method improves the QoE, it cannot conveniently respond to the typical use cases of many HAS servers. Moreover, it requires an additional processing task, which becomes burdensome when many HAS clients are demanding video contents from the same HAS server.
- The gateway-based solution consists of employing a bandwidth manager in the gateway that divides the available bandwidth between the HAS clients. The bandwidth manager defines a shaping rate for each HAS client based on the manifest files that define the available quality levels for each HAS session. Then, this manager shapes the sending bitrate for each client. This solution is more convenient than client-based and server-based solutions because the gateway can acquire information about the HAS traffic of all clients of the same home network, which is not possible for the server or the client. Additionally, the gateway-based solution does not require changes in the implementations of the player or the server.

One of the recent gateway-based solutions was proposed in the HTB shaping Method (HTBM) [6]. It employs the bandwidth manager with a specific shaping method, the "Hierarchical Token Bucket" (HTB) queuing discipline. It uses one link, designated as the parent class, to emulate several slower links, designated as the children classes, and to send different kinds of traffic on different emulated links. HTB also employs the "*tokens and buckets*" concept together with the class-based system, to achieve better control over traffic and for shaping in particular [14]. A fundamental part of the HTB queuing discipline is the "borrowing mechanism": children classes borrow tokens from their parent once they have exceeded the shaping rate. A child class will continue to try to borrow until it reaches a defined threshold of shaping, at which point it will begin to queue packets that will be transmitted when more tokens become available. HTBM considers each HAS stream as a child class. Accordingly, it enables shaping by delaying packets received from a HAS server. The authors of [6] indicate that HTBM improves the user's QoE; features that are enhanced include stability of video quality level, fidelity to optimal quality level, and convergence speed.

Our proposed method, Receive Window Tuning Method (RWTM) [13], was also integrated in the gateway due to the convenience of gateway-based solutions, described above. It is based on TCP flow control to define the shaping rate for each HAS client in accordance with the bandwidth manager's directions. Knowing that the sending rate of a given TCP session is bounded by the amount *min(rwnd, cwnd)/RTT*, where *rwnd* is the receiver's advertised window, *cwnd* is the congestion window of the sender, and *RTT* is the round trip time between the sender and the receiver, RWTM consists of limiting the *rwnd* of the HAS client so that the shaping rate of the bandwidth manager will be equal to *rwnd/RTT*. The methodology of RWTM involves acquiring the shaping rate value, *SHR*, from the bandwidth manager, to estimate the *RTT* value and to modify the *rwnd* of each HAS client in the gateway to be equal to *SHR*×*RTT*. The modification of *rwnd* is ensured in the gateway by modifying the *rwnd* field of each TCP acknowledgment (ACK) packet sent from a HAS client to the HAS server. The modification of *rwnd* and the estimation of *RTT* will be described in detail in Subsect. III-B. The investigations of [13] have demonstrated that RWTM improves QoE. The objective of this paper is to compare the two gateway-based shaping methods, HTBM and RWTM, by using an identical well-chosen bandwidth manager that is intended to define the optimal quality level for each HAS client.

III. Bandwidth management algorith and RWTM description

In this section, we describe the bandwidth manager that defines the shaping rate for HAS clients in accordance with the optimal quality level definition used for QoE metrics in Section I. We next provide a more accurate description of the RWTM method. For this purpose, we define in Table I the parameters that are used in this section.

TABLE I. DESCRIPTION OF PARAMETERS

Parameter	Description
avail_bw	Available bandwidth in home network for HAS streams
N	Total number of active HAS clients connected to the home network
C	HAS client index, C ϵ *{1,…,N}*
l	Video quality level index, or profile index
R_l	Rate of the l^{th} profile index; 10% higher than the profile encoding rate
$SHR_{C\text{-}S}$	The shaping rate for the HAS stream requested by the client C from the server S
$RTT_{C\text{-}S}$	The round-trip time between client C and server *S*
$rwnd_{C\text{-}S}$	The *rwnd* value defined by RWTM in the TCP ACK packet sent from client C to server S

A. *Bandwidth management algorithm*

The bandwidth manager is the fundamental component of the gateway-based solution. It uses as inputs three parameters: the *avail_bw*, *N,* and the manifest file of each HAS stream that defines profile encoding rates. Then, it computes the shaping rate for each HAS client as indicated in equation (1):

$$SHR_{C-S} = \left\{ \max(R_l) \,\middle|\, R_l \le \frac{C}{N} \times avail\_bw - \sum_{j=1}^{C-1} SHR_{j-S} \right\} \quad (1)$$

$SHR_{C\text{-}S}$ is chosen in a manner that enables the players to select optimal quality levels by ensuring both the fairest share of *avail_bw* between HAS clients and the maximum use of *avail_bw*. This choice necessitates that if the fair share of *avail_bw* according to $\{max(R_l) \mid R_l \le avail\_bw/N\}$ involves *avail_bw* underutilization, and if the unused portion of *avail_bw* enables one or more HAS clients to switch to a higher quality level, the bandwidth manager will allow it.

The bandwidth manager is sufficiently sophisticated to be able to update the number of active connected HAS clients in the home gateway, *N*, by sniffing the *SYN* and *FIN* flags in the headers of TCP packets. We also assume that it is capable of estimating the *avail_bw* and updating its value over time. Accordingly, the manager updates the shaping rate when any change occurs.

B. *Receive Window Tuning Method description*

1) *Constant RTT*

When $RTT_{C\text{-}S}$ is stationary, the definition of $rwnd_{C\text{-}S}$ will depend only on $SHR_{C\text{-}S}$ after the first estimation of $RTT_{C\text{-}S}$. Accordingly, computing $RTT_{C\text{-}S}$ in the three-way handshake phase of the TCP connection before data transfer, as proposed in [9], will be sufficient for RWTM processing. However, the authors of SABRE [7] reveal that during the ON period of a HAS stream, the $RTT_{C\text{-}S}$ value increases. This is caused by the queuing delay in the home gateway. In fact, the reception of many bursts of video packets fills the queue at the gateway, which causes new packets to experience long delays until the queue is drained. Thus, in order to improve the estimation of $RTT_{C\text{-}S}$ during the ON period, we multiply it by a weight empirical constant μ (μ >1) as shown in equation (2):

$$RTT^*_{C\text{-}S} = \mu \times RTT_{C\text{-}S} \qquad (2)$$

Hence, an improved definition of $rwnd_{C\text{-}S}$ is proposed in equation (3):

$$rwnd_{C\text{-}S} = SHR_{C\text{-}S} \times RTT^*_{C\text{-}S} \qquad (3)$$

2) *General case: variable RTT*

When $RTT_{C\text{-}S}$ becomes variable over time, RWTM should update the $rwnd_{C\text{-}S}$ value to maintain the same shaping rate as described in equation (4):

$$rwnd_{C\text{-}S}(t) = SHR_{C\text{-}S} \times RTT^*_{C\text{-}S}(t) \qquad (4)$$

The gateway may have to compute the $RTT_{C\text{-}S}$ value exhaustively. Since this may be a heavy processing task, our idea consists of only estimating $RTT_{C\text{-}S}$ from packets sent by the HAS client: this is called *passive estimation of RTT* [10]. As a result, we define two parameters, $RTT_{G\text{-}S}$ and $RTT_{C\text{-}G}$, which are the round trip time between the home gateway and the HAS server, and the round trip time between the HAS client and the gateway, respectively. Since all packets circulating between the HAS client and HAS server pass through the home gateway, we can provide the following equation:

$$RTT_{C\text{-}S} \approx RTT_{C\text{-}G} + RTT_{G\text{-}S} \qquad (5)$$

$RTT_{C\text{-}S}$ estimation is only possible when the HAS client sends a HTTP GET request message, as illustrated in Fig. 1. In fact, the HAS client will receive the HTTP response message after one $RTT_{C\text{-}S}$ and will immediately send an ACK packet. The time difference between the HTTP GET request message and the first ACK message in the home gateway is close to the $RTT_{C\text{-}S}$ value because it satisfies equation (5).

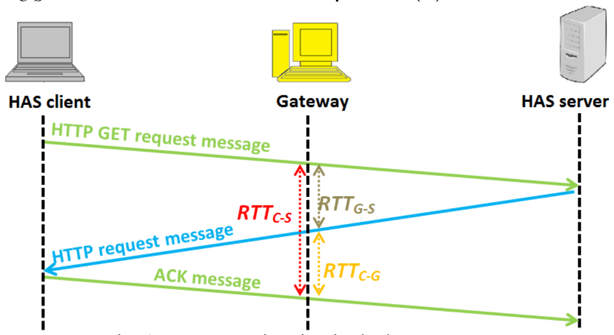

Fig. 1. $RTT_{C\text{-}S}$ estimation in the home gateway

Accordingly, the home gateway will be able to estimate $RTT_{C\text{-}S}$ at the beginning of each chunk request. Then, it applies equations (2) and (4) to update the $rwnd_{C\text{-}S}$ value.

IV. Experimental Implementation

We propose a testbed architecture, presented in Fig. 2, that emulates our use case described in Section I. The choice of only two clients is sufficient to demonstrate the behavior of the concurrence between many HAS flows in the same home network. In this paragraph, we describe the configurations of each component presented in Fig. 2:

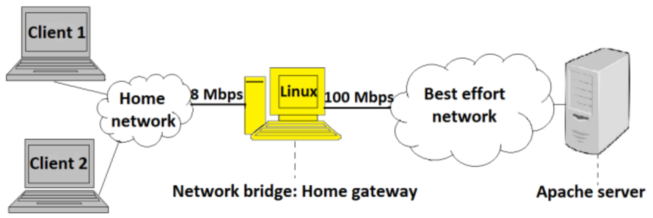

Fig. 2. Architecture used in testbed

- HAS server

The HAS server is modeled by an HTTP/1.1 Apache Server installed on a Linux machine operating on Debian version 3.2. It employs CUBIC [11] as its TCP congestion control algorithm. All tests use five video quality levels denoted by 0, 1, 2, 3 and 4. Their encoding rates are constant and equal to 248 kbps, 456 kbps, 928 kbps, 1,632 kbps, and 4,256 kbps, respectively. The playback duration of a chunk is 2 seconds.

- Best-effort network

The best-effort network is characterized by the presence of network devices to route packets. The round trip time $RTT_{C\text{-}S}(t)$ in a best-effort network is modeled by equation (6) in [8] as follows:

$$RTT_{C\text{-}S}(t) = a_{C\text{-}S} + q(t)/\varsigma \qquad (6)$$

where $a_{C\text{-}S}$ is a fixed propagation delay between client C and server S; $q(t)$ is the queue length of a single congested router, which is the home gateway in our use case; and ς is the transmission capacity of the router. Indeed, $q(t)/\varsigma$ models the queuing processing delay.

To comply with equation (6), we used the normal distribution with a mean value $a_{C\text{-}S}$ and a standard deviation equal to $0.07.a_{C\text{-}S}$. The standard deviation emulates the queuing processing delay $q(t)/\varsigma$. This emulation is accomplished by using the "*netem delay*" parameter of the *traffic controller* tool in the gateway machine interfaces.

- Home gateway

We consider the two connected components, DSL router and DSLAM, as unique equipment called the home gateway. The emulated home gateway consists of a Linux machine configured as a network bridge to forward packets between the home network and the best-effort network. We emulate the queuing discipline of DSLAM by using two 64-packet-size FIFO subqueues. In the gateway, we implemented the bandwidth manager as described in Subsect III-A. The two shaping methods, HTBM and RWTM, are implemented in this gateway, and they shape bandwidth in accordance with the bandwidth manager's decisions, as described in Sections II and III. The weight constant μ of equation (2) employed by RWTM is chosen empirically and is equal to $\mu = 1.275$.

- Home network

In the modeled home network, the clients are connected to the gateway. The bandwidth is limited to 8 Mbps. We selected this value because it is lower than twice the video encoding bitrate of the highest quality level. As a result, two clients in the home network cannot easily select the highest quality level at the same time. In this case, client 1 should select quality level 4 and client 2 should select quality level 3 as the optimal qualities defined by the bandwidth manager. We do not employ a use case in which two clients can select the same quality level, because this is a specific case in reality, and dissimilarity between optimal quality levels represents a more general case.

- HAS clients

We used two Linux machines as HAS clients. We developed a simple player in each client that reproduces the behavior of the HAS player without decoding and displaying a video stream.

V. EVALUATION OF THE RESULTS

In this section, we evaluate the QoE between HTBM and RWTM. Accordingly, we define objective QoE metrics and three significant scenarios.

A. *QoE metrics*

In this paragraph, we define analytically three objective QoE metrics in accordance with Section I. For this purpose, we define in Table II the parameters to be used in this section.

TABLE II. PARAMETER DESCRIPTION

Parameter	Description
i	Discrete time index
$L_C(i)$	Video quality level index of client C at time i $L_C(i) \in \{0,1,2,3,4\}$
$L_{C,opt}(i)$	Theoretical optimal value of $L_C(i)$
$Q_C(i)$	Video encoding bitrate of client C at time i

1) Instability of video quality level

We employ a defined instability metric of FESTIVE [4], $IS_C(K)$; it measures the instability rate of client C for a test duration of K seconds:

$$IS_C(K) = \frac{\sum_{i=0}^{K-1} |Q_C(K-i) - Q_C(K-i-1)| \times w(i)}{\sum_{i=1}^{K} Q_C(K-i) \times w(i)} \quad (7)$$

where $w(i) = K\text{-}i$ is a weight function that adds a linear penalty to a more recent quality level switch.

2) Infidelity to optimal quality level selection

We define the infidelity metric $IF_C(k)$ of client C for a test duration of K seconds. It measures the portion of time during which the HAS client C requests optimal quality.

$$IF_C(K) = \left\{ \frac{\sum_{i=1}^{K} \theta_C(i)}{K} \middle| \theta_C(i) = \begin{cases} 1, if\ L_C(i) \neq L_{C,opt}(i) \\ 0, else \end{cases} \right\} \quad (8)$$

3) Convergence speed

We use the convergence speed metric described in [6]. It is defined as follows:

$$V_{C,T}(K) = \left\{ \min_{d \in \{1,\dots K\}} (d) \middle| L_C(i) = L_{C,opt}(i)\ \forall\, i \in [d, d+T] \right\} \quad (9)$$

This metric describes the duration of time that the player of HAS client C requires to reach a stable optimal quality level for at least T seconds for a test duration of K seconds.

B. *Scenarios*

We define three scenarios that describe the concurrence between HAS clients in the home network. We used only two HAS clients because this setup is easy to analyze and is sufficient to show the behavior of the competition for *avail_bw*:

1. Both clients start to play simultaneously and continue for 3 minutes. This scenario shows how clients compete.
2. Client 1 starts to play, and after 30 seconds, the second client starts; both continue together for 150 seconds. This scenario shows how a transition from one client to two clients occurs.
3. Both clients start to play simultaneously, and after 30 seconds, we stop client 2; client 1 continues alone for 150 seconds. This scenario shows how a transition from two clients to one takes place.

The choice of test duration of 3 minutes has an objective to offer sufficient delay for players to stabilize.

C. *Results analysis*

In this section, we discuss the QoE measurements of the two shaping methods during the three scenarios, and we provide a more precise explanation of the observed results by showing the congestion window variation.

1) Scenarios 1, 2 and 3

For each scenario, we repeated each test 60 times, and we employed the average value of QoE metrics in our evaluation. The number of 60 runs was justified by the fact that the difference of the average results obtained after 40 runs and 60 runs is lower than 6%. This observation was verified for all scenarios. Accordingly, using 60 tests is sufficient to yield statistically significant results. The average values of metrics are listed in Table III. We note that convergence speed metric is computed for scenarios 2 and 3 from 30 seconds instead of the beginning of the test, i.e., when the number of HAS clients changes.

TABLE III. PERFORMANCE METRIC MEASUREMENTS AVERAGE VALUES

Scenario		1		2	3	Average
Client		*1*	*2*	*1*	*1*	*1*
Instability (%) **$IS_c(k)$**	***W/o****	7.47	5.82	3.87	2.18	4.5
	HTBM	1.86	1.15	3.44	2.19	2.49
	RWTM	1.63	1.13	1.43	1.63	1.56
Infidelity (%) **$IF_C(k)$**	***W/o***	50.46	36.93	67.36	13.94	43.92
	HTBM	20.45	4.47	32.09	18.49	23.95
	RWTM	5.02	2.61	3.42	4.81	4.42
Convergence speed (seconds) **$V_{C,60}(k)$**	***W/o***	DC**	92.81	DC	20.1	-----
	HTBM	52.06	13.26	64.13	34.65	50.28
	RWTM	19.55	8.95	10.98	14.36	14.96

*Without Shaping **Did not converge

For all scenarios, we verified that the two shaping methods improved the QoE, except as noted. For the first scenario, the simultaneous playing of two clients involves better QoE measurements for client1 than client2. This behavior is expected because the bandwidth manager assigns lower optimal quality level (n° 3) for client 2 than client 1 (n° 4), which is easier to achieve. Moreover, we observe that RWTM has better QoE than HTBM, and the gap between measurements is higher for client 1. RWTM is 4 times more faithful to optimal quality level (*IF*=5.02% vs. 20.45%) and converges 2.6 times faster (*V*=19.55 s vs. 52.06 s) than HTBM.

For the second scenario, we observe an improvement of QoE measurements of RWTM but a degradation of HTBM. This finding suggests that HTBM is highly disturbed by the increase of HAS clients from one to two clients.

For the third scenario, we observe that the instability measurements of the three methods are similar. This result is expected because the client operates alone for 150 seconds, and therefore is easier to stabilize. Moreover, HTBM has the worst instability and convergence speed measurement. Additionally, RWTM maintains practically the same QoE measurements of scenario 1.

The last column of Table III indicates the average performance values of three scenarios related to client 1. We can conclude that RWTM is 37.3% more stable (*IS*=1.56% vs. 2.49%), 5.41 times more faithful to optimal quality level (IF=4.42% vs. 23.95%), and has convergence that is 3.4 times faster (V=14.96 s vs. 50.28 s) than HTBM. We can conclude that the recommended gateway-based method to be used for improved user's experience is RWTM. We provide an explanation of these results in the next subsection.

2) *Congestion window variation*

In order to explain the cause of these results, we used the *tcp_probe* module in the HAS server. This module shows the evolution of the congestion window, *cwnd*, and slow start threshold, *ssthresh*, during each test. We selected two tests that have the nearest QoE measurements of the first column of Table III related to HTBM and RWTM, respectively. We present the *cwnd* and *ssthresh* variation of the two tests in Fig. 3 and 4. In the two figures, we indicate by a vertical bold dotted line the moment of convergence. We observe that this moment coincides with a considerably higher stability of *ssthresh* and higher stability of *cwnd* in the congestion avoidance phase.

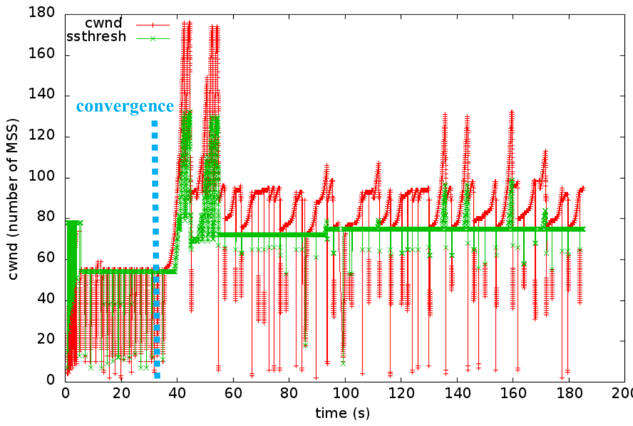

Fig. 3. *Cwnd* and *ssthresh* variation when using HTBM (*IS*=1.98% , *IF*=19.03%, *V*=33 s)

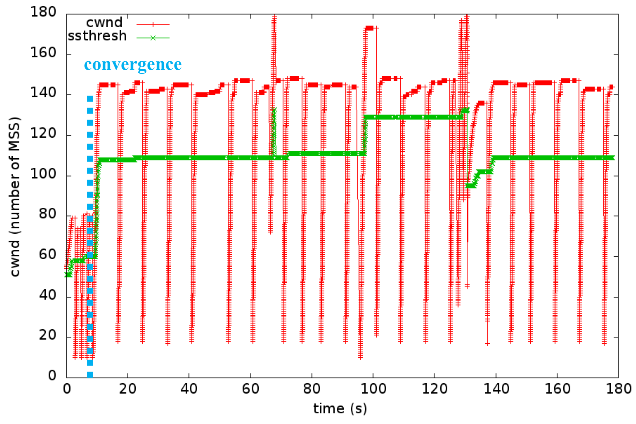

Fig. 4. *Cwnd* and *ssthresh* variation when using RWTM (*IS*=1.78%, *IF*=5.5%, *V*=8 s)

In order to be accurate in our analyses, based on the CUBIC [11] congestion control algorithm used in the server and its source code *tcp_cubic.c,* we present some important value updates of *cwnd* and *ssthresh* for different events:

- Congestion event - we have two cases:
  - When three duplicated ACKs are received, the Fast Recovery / Fast Retransmit (FR/FR) phase reduces *ssthresh* and sets *cwnd* to *ssthresh+3*. Hence, c*wnd* remains in congestion avoidance phase.
  - When the retransmission timeout expires before receiving ACK of the lost packet, *ssthresh* is reduced, and *cwnd* is set to a small value and restarts from slow start phase.
- Idle period: When the server sends a packet after an idle period that exeeds retransmission timeout (*RTO*), *ssthresh* is set to *max(ssthresh, ¾ cwnd)*, and the *cwnd* value is computed as in Algorithm 1:

Algorithm 1 *cwnd* update after an idle period

```
1: for i=1 to int(idle/RTO) do
2:     cwnd = max ( min ( cwnd , rwnd )/2, 1 MSS )
3: end for
```

where *MSS* is the maximum segment size.

In the context of HAS, an idle period coincides with an OFF period between two consecutive chunks. We denote by *OFF** the OFF period whose duration exceeds *RTO*.

After convergence, with HTBM we have a high congestion rate: as presented in Fig. 3, the majority of congestions is caused by retransmission timeout. In contrast, with RWTM, congestion events are negligeable: only two visible congestions occur in Fig. 4. As a consequence, *ssthresh* becomes lower when using HTBM than when using RWTM: 75 *MSS* in Fig. 3 vs. 110 *MSS* in Fig. 4. On the other hand, RWTM has frequent *OFF** periods, with one *OFF** period every 4.14 chunks, on average, unlike HTBM, which shows pratically no idle periods in Fig. 3. In Fig. 4, *cwnd* is divided by 8 after each *OFF** period. Based on Algorithm 1, the *OFF** period's duration is included in the interval *[3 RTO, 4 RTO]*. This observation leads us to compute the *RTT* variation between client 1 and server, $RTT_{1\text{-}S}$, for each test. $RTT_{1\text{-}S}$ jumps from 100 ms to around 200 ms when using HTBM, as presented in black in Fig. 5. In contrast, when using RWTM, $RTT_{1\text{-}S}$ increases slightly and reaches approximately 120 ms, as presented in green in Fig. 5.

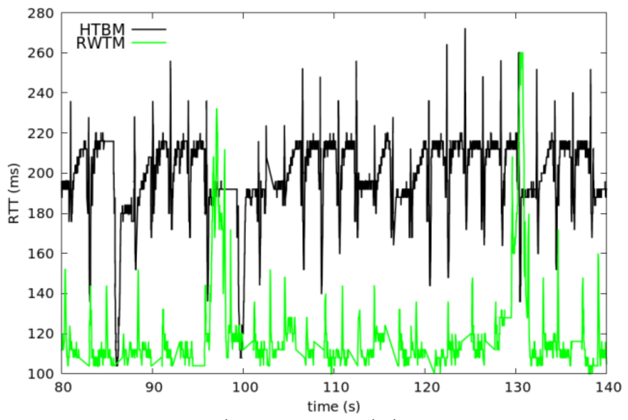

Fig. 5. $RTT_{1\text{-}S}$ variation

Consequently, approximately 100 ms of queuing delay is generated by HTBM, in contrast to about 20 ms generated by RWTM. As result, on one hand, a large number of packets is buffered in the gateway queue when using HTBM, and that increases the congestion rate. On the other hand, the high queuing delay generated by HTBM has the advantage of reducing the frequency of *OFF** periods by increasing *RTO* duration: the ratio (*OFF*)/RTO* is reduced. Even if an *OFF** period occurs, based on Algorithm 1, the *cwnd* value will be divided, but will still be twice as high as when using RWTM.

So, on one hand, HTBM delays packets considerably, which causes a high congestion rate and, by consequence, lower convergence speed, but it practically eliminates *OFF** periods. On the other hand, RWTM does not generate congestions, but concerning *OFF** periods, RWTM reduces their frequency but does not eliminate them. The cause is the non-exhaustive estimation of $RTT_{C\text{-}S}$, i.e. limited to only one estimation per chunk, which may induce a low conformity of the shaping rate computation to the $SHR_{C\text{-}S}$ value defined by the bandwidth manager.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have conducted a comparative evaluation between two gateway-based shaping methods, HTBM and RWTM, employed to improve the HAS user's experience (QoE). The use case is defined as HAS clients being located in the same home network and competing for bandwidth. We define the same bandwidth manager in the gateway, and we present the testbed and its parameters to permit an accurate comparison. By defining objective QoE criteria, the comparative evaluation shows that RWTM is more beneficial; it is 37.3% more stable, 5.41 times more faithful to optimal quality level, and converges 3.4 times faster to the optimal quality level than HTBM. The main explanation of this result is directly related to the additional queuing delay induced by HTBM to shape HAS traffic, while RWTM just reduces the advertised window of HAS clients and thus does not add significant queueing delay.

However, RWTM estimates the round trip time between the client and the server, $RTT_{C\text{-}S}$, once for every chunk. This leads us to inquire about the robustness of the shaping method against $RTT_{C\text{-}S}$ instability in wireless home networks. This is a very valuable aspect of our research that should be investigated in future work.

REFERENCES

[1] W. Van Lancker et al. "HTTP Adaptive Streaming with Media Fragment URIs". IEEE Intl. Conf. on Multimedia and Expo (ICME), 2011.

[2] S. Akhshabi, et al.. "What Happens when HTTP Adaptive Streaming Players Compete for Bandwidth?." ACM workshop on Network and Operating System Support for Digital Audio and Video, 2012.

[3] BJ. Villa and PE. Heegaard. "Group Based Traffic Shaping for Adaptive HTTP Video Streaming by Segment Duration Control." Advanced Information Networking and Applications (AINA), 2013.

[4] J. Jiang, V. Sekar, and H. Zhang. "Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming with Festive," IEEE/ACM transactions on networking, VOL. 22, NO . 1, Feb 2014.

[5] S. Akhshabi, et al. "Server-based Traffic Shaping for Stabilizing Oscillating Adaptive Streaming Players." ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, 2013.

[6] R. Houdaille and S. Gouache. "Shaping HTTP Adaptive Streams for a Better User Experience." ACM Multimedia Systems Conference, 2012.

[7] A. Mansy, B. Ver Steeg, and M. Ammar. "SABRE: A Client Based Technique for Mitigating the Buffer Bloat Effect of Adaptive Video Flows." The 4th ACM Multimedia Systems Conference, 2013.

[8] V. Misra, W. Gong, and D. Towsley. "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED." In ACM SIGCOMM Computer Communication Review, 2000

[9] S. Wan and Y. Hao. "Research on TCP Optimization Strategy of Application Delivery Network", IEEE Intelligent Computation and Bio-Medical Instrumentation (ICBMI), 2011

[10] H. Jiang and C. Dovrolis "Passive Estimation of TCP Round-trip Times." ACM SIGCOMM Computer Communication Review 32, 2002.

[11] S. Han I Rhee and L Xu. "CUBIC: a New TCP-friendly High-speed TCP variant." ACM SIGOPS Operating Systems Review 42, 2008.

[12] L. Stewart et al. "Multimedia-unfriendly TCP Congestion Control and Home Gateway Queue Management," ACM conference on Multimedia systems, 2011.

[13] C. Ben Ameur, E. Mory and B. Cousin, "Shaping HTTP Adaptive Streams Using Receive Window Tuning Method in Home Gateway", The 33rd IEEE International Performance Computing and Communications Conference (IPCCC). Dec. 2014.

[14] Martin A. Brown, Traffic Control HOWTO, at http://linux-ip.net/articles/Traffic-Control-HOWTO. 2006.