Content Delivery Networks
Interconnection Working Group
Internet-Draft
Intended status: Informational
Expires: July 28, 2013

J. Famaey
S. Latre
UGent - iMinds

January 24, 2013

Experiments on HTTP Adaptive Streaming over interconnected Content Delivery Networks
draft-famaey-cdni-has-experiments-01

Abstract

This document reports experimental results on the delivery of HTTP Adaptive Streaming (HAS) content over interconnected Content Delivery Networks (CDNs). Specifically, the implications that CDN request routing between CDNs and HTTP redirection have on the quality of delivered HAS content are investigated.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 28, 2013.

Table of Contents

1. Introduction

HTTP Adaptive Streaming (HAS) refers to a set of novel streaming approaches, which deliver streaming media content over the HTTP protocol. The content is split into chunks, offered in several quality layers. This allows the client to dynamically adapt the requested quality, based on available network resources and device capabilities. Delivering HAS content across multiple interconnected CDNs introduces some new opportunities and challenges. Specifically, it becomes possible to distribute the chunks of a single HAS content stream across servers deployed on multiple CDNs, based on chunk popularity, Quality of Service requirements, resource availability, costs or other factors.

Every HAS content stream is accompanied by a Manifest File, which lists the chunks of each quality layer and specifies their location in the form of a URL. As stated in [I-D.brandenburg-cdni-has], several alternative methods exist for specifying chunk locations:

- Relative URLs: The URLs specified in the Manifest File are relative to the Manifest File's location and thus all located on the same surrogate.

- Absolute URLs with Redirection: The Manifest File specifies the fully qualified URL of each chunk. These URLs, however, direct the client towards the CDN's request routing node, which in turn uses HTTP redirection to send the client to the surrogate hosting the actual chunk.

- Absolute URLs without Redirection: The URL points directly to the surrogate hosting the chunk, effectively allowing the client to circumvent the CDN request routing process.

This document aims to evaluate and compare different request routing policies for HAS content, derived from these addressing mechanisms, that can be used in federated CDN scenarios.

2. Experimental Setup

The scenario used as a basis for the experiments consists of two interconnected CDNs. The downstream CDN is located close to the end-user (e.g., a telco CDN), while the upstream CDN is positioned further (e.g., in the core Internet). The upstream CDN is assumed to be the main storage facility of the original content. As such, it hosts the Manifest file but can offload content chunks to one or more downstream CDNs. Figure 1 graphically depicts the scenario and lists the parameters that were varied in the course of the experiments.

The upstream CDN request router, upstream CDN content server, downstream CDN request router and downstream CDN content server are depicted as uRR, uCS, dRR and dCS, respectively. During the experiments, five parameters were varied: the one-way Internet delay ID, the one-way downstream CDN delay DD, the per-client bandwidth B, the HAS client buffer size P and the HAS segment duration S. The bandwidth on all other network links was set to 100 Mbps, while the one-way network delay was set to 5 ms. The round trip time (RTT) between two nodes can be calculated as the sum of the one-way delays of the links on the path between them, multiplied by two. In the performed experiments, the client and dRR/dCS are separated by three links, resulting in a RTT of (2 * 5 + DD) * 2 = (20 + 2 DD) ms. On the other hand, the client and uRR/uCS are 5 links apart, resulting in a RTT of (4 * 5 + ID) * 2 = (40 + 2 ID) ms. Note that the figure presents a high level, simplified view of the network topology and does not show all individual network links and routers. Additionally, the processing delay on the CDN surrogates is not taken into account, as it is assumed to be negligible compared to the network delay.

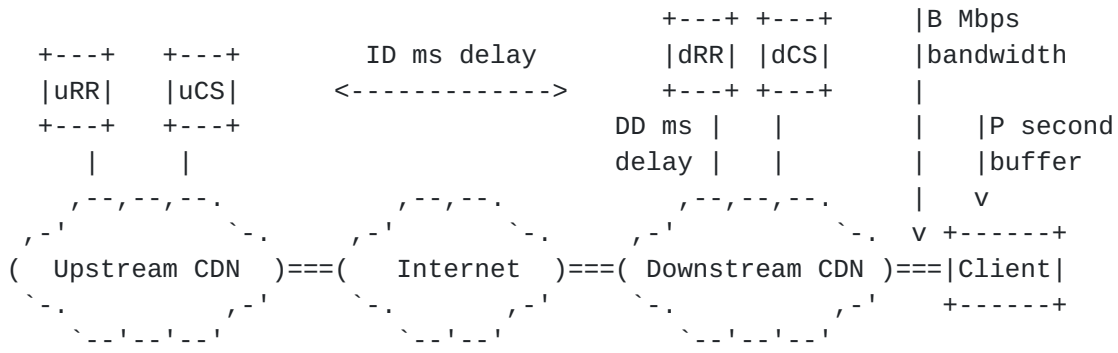

Figure 1: Evaluation scenario and parameters

Three alternative request routing policies are evaluated and compared:

- UpstreamRR: The Manifest File points to the uRR for every chunk. If the chunk is located within the upstream CDN's network, the uRR sends the client a HTTP redirect request to point it to the correct uCS. Otherwise, the uRR redirects the client to dRR, which in turn redirects it to the correct dCS.

- DirectRR: The Manifest File immediately points to the correct request router, which redirects the client to the correct content server. This policy thus allows the client to circumvent going via the upstream CDN's network if the chunk is located downstream.

- DirectCS: The Manifest File immediately points to the correct content server, which allows the client to download segments without being redirected. Compared to the DirectRR policy, the indirection of contacting the dRR is avoided.

The UpstreamRR policy can be seen as the traditional CDN-I approach, where clients always contact the original CDN and HTTP redirection is used to point them to interconnected CDNs when necessary. It does not require any Manifest File rewriting. Additionally, the upstream CDN does not need any detailed information about chunk locations, as it only needs to redirect clients to the downstream request router. The DirectRR and DirectCS policies are more complex, as they require the upstream CDN to rewrite the original Manifest File. Additionally, when using the DirectCS policy, the downstream CDN either needs to share detailed chunk location information with the upstream CDN or the interconnected CDNs need to collaborate in creating the Manifest File.

The experiments evaluate a scenario where a single client downloads a 200 second video clip (split into 200/S segments). The first half is hosted by the downstream CDN, while the second half is hosted by the upstream CDN. The constant bitrate (CBR) video is available in 3 qualities, with bitrates 500 kbps, 1 Mbps and 2 Mbps respectively.

As the end-user Quality of Experience (QoE) depends on several factors, multiple evaluation metrics are used in the comparison:

- Average played quality: The played quality layer, averaged over all chunks and specified in terms of bitrate. This is expressed in megabits per second (Mbps), representing the bandwidth required for downloading the played quality layers.

- Total buffer starvation time: The accumulated time during which the client needs to rebuffer the chunks (excluding the original start-up). A rebuffering occurs when a chunk is not available at the client, while it is already required for decoding. This leads to frame freezes, as the client needs to wait for the next chunk to arrive, which significantly reduces QoE.

- Start-up delay: The time between the initial HTTP request for the first chunk, performed by the client, and the time when the chunk actually starts playing.

All reported results were obtained using the NS-3 simulation environment [ns3] in combination with the Network Simulation Cradle (NSC) [nsc]. NS-3 is a discrete-event network simulator for Internet systems. NSC allows NS-3 to interface directly with the kernel's TCP implementation, generating more accurate and realistic results. The

used HAS client rate adaptation algorithm is based on the first version of the client algorithm incorporated in Microsoft's SmoothStreaming client. The source code of this algorithm can be retrieved from CodePlex [msscode].

3. Results

This section lists and discusses experimental results on the average played video quality, total buffer starvation time and the start-up delay. First, the effects of several parameters on the QoE metrics are studied, both in a congested and an uncongested network. Second, the influence of segment duration is evaluated.

3.1. Congested Scenario

The congested scenario considers a client-side bandwidth B of 1Mbps, which, due to protocol overhead allows only the lowest 500kbps quality to be streamed. As such, this section focuses on a comparison of the buffer starvation time and start-up delay. The segment duration S is fixed at 2s. The results on buffer starvation as a function of one-way Internet delay ID, one-way downstream CDN delay DD and client buffer size P are shown in Figure 2. The starvation time is shown separately for the first 50 segments downloaded from dCS (Dws) and the latter 50 downloaded from uCS (Ups). The results on start-up delay are depicted in Figure 3 as a function of delays ID and DD only, as they are unaffected by the buffer size P.

In general, the results depicted in Figure 2 clearly show that minimising the number of HTTP redirects benefits the QoE significantly, both for segments hosted at the downstream as well as the upstream CDN. Specifically, several observations can be made based on the depicted results. First, as expected, DirectCS and DirectRR are not influenced by an increasing Internet delay for downstream segments, as they completely circumvent the upstream CDN in this case. In contrast, when using the traditional UpstreamRR approach buffer starvations start occurring at a one-way Internet delay of as low as 100ms if the downstream CDN delay is 50ms or higher. If the downstream delay is low, then starvations start occurring at a 200ms Internet delay. Second, for upstream segments, DirectRR and DirectCS are also negatively impacted by an increasing Internet delay. However, DirectCS is significantly less influenced by it than DirectRR, as it circumvents DirectRR's redirect from uRR to uCS. Third, increasing the buffer size clearly helps reducing buffer starvations in the upstream scenario for DirectRR and DirectCS. This is due to the fact that the client can fill its buffer when downloading the first 50 segments from dCS. This set of

backup segments subsequently allows the client to temporarily maintain desirable QoE levels under high RTT. When using UpstreamRR, the client cannot fill its buffer during the initial phase, due to the larger number of redirects, and a larger buffer therefore does not improve results.

P (s)	DD (ms)	ID (ms)	Total buffer starvation time (s)					
			UpstreamRR		DirectRR		DirectCS	
			Dws	Ups	Dws	Ups	Dws	Ups
6	5	50	0.0	0.0	0.0	0.0	0.0	0.0
		100	0.0	0.1	0.0	0.0	0.0	0.0
		200	10.3	34.4	0.0	30.4	0.0	10.2
	50	50	0.0	0.0	0.0	0.0	0.0	0.0
		100	5.5	3.8	0.0	0.0	0.0	0.0
		200	18.6	34.5	0.0	30.4	0.0	10.2
24	5	50	0.0	0.0	0.0	0.0	0.0	0.0
		100	0.0	0.0	0.0	0.0	0.0	0.0
		200	10.4	34.4	0.0	12.4	0.0	0.0
	50	50	0.0	0.0	0.0	0.0	0.0	0.0
		100	5.5	3.8	0.0	0.0	0.0	0.0
		200	18.6	34.4	0.0	13.5	0.0	0.0

Figure 2: The total buffer starvation time as a function of client buffer size P and one-way Internet delay ID and one-way downstream CDN delay DD; for B = 1Mbps and S = 2s

ID (ms)	DD (ms)	Start-up delay (s)		
		UpstreamRR	DirectRR	DirectCS
50	5	2.11	1.81	1.72
	50	2.92	2.63	2.37
100	5	2.31	1.81	1.72
	50	3.12	2.63	2.37
200	5	2.71	1.81	1.72
	50	3.52	2.63	2.37

Figure 3: The start-up delay as a function of one-way Internet delay ID and one-way downstream CDN delay DD; for B = 1Mbps, S = 2s and P = 6s

The results in Figure 3 clearly show that the start-up delay is linearly proportional to the delay between the client and server. This explains the two evolutions visible in the table. First, for segments hosted at the downstream CDN, the start-up delay for UpstreamRR increases as a function of the one-way Internet delay ID, while DirectRR and DirectCS are unaffected. Second, the start-up delay for all routing policies increases as a function of the one-way downstream delay DD. Finally, due to the lower redirection delay of DirectCS compared to DirectRR, the DirectCS start-up delay is slightly lower. Note that this start-up delay occurs whenever the buffer needs to be flushed. As such, this not only happens when a client initiates a session, but also for example when switching channels in Internet TV scenarios or skipping to another part of a movie in a Video on Demand scenario.

In summary, it was shown that in congested scenarios, using the DirectRR or DirectCS routing policies can significantly reduce the amount of client-side buffer starvation compared to using the traditional UpstreamRR policy, when downloading HAS segments from the downstream CDN. Additionally, the use of DirectCS is beneficial in terms of buffer starvations compared to DirectRR and UpstreamRR when downloading segments from the upstream CDN. Although a larger buffer size was shown to help in temporarily overcoming the negative effects of high network latency, this only helps if a client can first fill its buffer by downloading segments with low latency. Finally, it was shown that the start-up delay is linearly proportional to the total

RTT, both caused by network latency to the content server, as well as redirection delay. As such, the DirectRR and DirectCS start-up delay is unaffected by the Internet delay when streaming from the downstream CDN, while that of UpstreamRR is not. Moreover, DirectCS has a lower start-up delay than DirectRR.

3.2. Uncongested Scenario

The uncongested scenario considers a client-side bandwidth B of 5Mbps, which is sufficient to download the highest 2Mbps quality layer stream. As such, this section does consider the delivered video quality. As buffer starvation and start-up delay results were already discussed in much detail in the previous section, and they show similar trends in the uncongested scenario, they are omitted here. The segment duration S is once again fixed at 2s. The results on average played video quality as a function of one-way Internet delay ID, one-way downstream CDN delay DD and client buffer size P are shown in Figure 4. The quality is shown separately for the first 50 segments downloaded from dCS (Dws) and the latter 50 downloaded from uCS (Ups).

The results in Figure 4 prove that an increased number of redirections significantly impacts video quality, even for a relatively low RTT. Specifically, the results show that UpstreamRR achieves a significantly lower quality than DirectRR and DirectCS for segments hosted at the downstream CDN for all depicted parameter combinations. Additionally, as expected, the delivered video quality when using UpstreamRR is inversely proportional to the Internet delay ID. In contrast, DirectRR and DirectCS are unaffected. In addition to the quality difference for segments hosted at the downstream CDN, there also are some remarkable differences for upstream CDN segments. Although UpstreamRR and DirectRR exhibit the same behaviour when downloading segments from the upstream CDN, they do show a difference in video quality. This is due to suboptimal decision making by the MSS client algorithm when switching servers. The UpstreamRR policy often results in a lower quality being requested for the downstream segments. However, when switching to the upstream CDN, the algorithm might not always decide to increase the quality, even if this would in theory be possible. This behaviour is caused by the way clients estimate the available throughput, which does not take into account multiple servers. In contrast, when using DirectRR the client is already downloading a higher quality from the dCDN, and will not change this when switching to the uCDN. Consequently, suboptimal decisions of the client algorithm, as a consequence of server switching, can lead to unexpected differences between UpstreamRR and DirectRR. Finally, the results show that increasing the buffer size leads to a higher delivered video quality in almost all cases.

P (s)	DD (ms)	ID (ms)	Average played quality (Mbps)					
			UpstreamRR		DirectRR		DirectCS	
			Dws	Ups	Dws	Ups	Dws	Ups
6	5	50	0.50	0.50	1.93	1.14	1.95	1.27
		100	0.50	0.50	1.93	1.02	1.95	1.03
		200	0.50	0.50	1.93	0.53	1.95	0.54
	50	50	0.50	0.50	0.97	1.00	0.98	1.00
		100	0.50	0.50	0.97	0.51	0.98	0.52
		200	0.50	0.50	0.97	0.51	0.98	0.52
24	5	50	1.64	2.00	1.93	2.00	1.95	2.00
		100	0.85	1.00	1.93	1.04	1.95	0.98
		200	0.50	0.50	1.93	0.69	1.95	0.66
	50	50	0.50	0.50	1.38	2.00	1.40	2.00
		100	0.50	0.50	1.38	1.01	1.40	0.98
		200	0.50	0.50	1.38	0.65	1.40	0.66

Figure 4: The average played quality as a function of client buffer size P, one-way Internet delay ID and one-way downstream CDN delay DD; for B = 5Mbps and S = 2s

In summary, the merits of DirectRR and DirectCS compared to UpstreamRR in uncongested networks were clearly shown. In addition to a reduction in buffer starvations and start-up delay, the DirectRR and DirectCS policies also result in an increased delivered video quality when enough bandwidth is available. Specifically, when using UpstreamRR and streaming content from the downstream CDN, the video quality is significantly impaired by an increase in Internet delay, while DirectRR and DirectCS are unaffected. Additionally, due to the fact that HAS client algorithms are unoptimised for delivery of a single stream from multiple servers, UpstreamRR additionally suffers a reduction in video quality compared to DirectRR for segments served from the upstream CDN in some scenarios.

3.3. Influence of Segment Duration

Previous sections only considered a short segment duration S of 2s. Although this value is widely used, by for example Microsoft SmoothStreaming-based services, others, such as Apple HTTP Live Streaming, generally recommend longer segment durations. This section evaluates the effect of segment duration S on the average played quality as well as the start-up delay in an uncongested scenario with a client-side bandwidth B of 5Mbps. As results showed that the used HAS algorithm performs poorly if the buffer fits only two segments or less and a segment duration of up to 12s is considered, a buffer size P of 36s is used. The results on average played quality as a function of segment duration S, one-way Internet delay ID and one-way downstream CDN delay DD are shown in Figure 5. The quality is shown separately for the first 50 segments downloaded from dCS (Dws) and the latter 50 downloaded from uCS (Ups). The results on start-up delay are depicted in Figure 6 as a function of S, ID and DD.

The results depicted in Figure 5 clearly prove that increasing the segment duration greatly improves performance of the three routing policies for large delays. If both ID and DD are large enough, it evens out performance of the three policies completely, removing the negative effects of redirects. This is obviously due to the fact that increasing the segment duration, decreases the number of requests and thus the relative delay introduced by redirects. On the other hand, longer segment durations result in lower average quality when the delay is very low (i.e., ID = 50ms, DD = 5ms). This is because increasing the segment duration results in a proportional increase in convergence time to the optimal video quality.

S (s)	DD (ms)	ID (ms)	Average played quality (Mbps)					
			UpstreamRR		DirectRR		DirectCS	
			Dws	Ups	Dws	Ups	Dws	Ups
2	5	50	1.95	2.00	1.93	2.00	1.95	2.00
		100	1.59	1.46	1.93	1.46	1.95	1.16
		200	1.15	0.75	1.93	0.74	1.95	0.72
	50	50	1.43	2.00	0.96	1.00	1.16	2.00
		100	0.81	1.00	0.96	1.00	1.16	1.16
		200	0.50	0.50	0.96	0.70	1.16	0.72
12	5	50	1.83	2.00	1.83	2.00	1.83	2.00
		100	1.72	2.00	1.83	2.00	1.83	2.00
		200	1.72	2.00	1.83	2.00	1.83	2.00
	50	50	1.61	2.00	1.61	2.00	1.61	2.00
		100	1.61	2.00	1.61	2.00	1.61	2.00
		200	1.61	2.00	1.61	2.00	1.61	2.00

Figure 5: The average played quality as a function of segment duration S, one-way Internet delay ID and one-way downstream CDN delay DD; for B = 5Mbps and P = 36s

Although using longer segment durations is an effective way to increase the video quality in face of redirects with high latency, it also has several disadvantages. As shown in Figure 6 the start-up delay increases significantly as a function of the segment duration. This is the case for all routing policies. Additionally, long segment durations are usually a poor choice in combination with live services, as they lead to significant lag of the streaming session compared to the live time.

S (s)	ID (ms)	DD (ms)	Start-up delay (s)		
			UpstreamRR	DirectRR	DirectCS
2	50	5	0.77	0.48	0.40
		50	1.56	1.26	1.00
	200	5	1.38	0.48	0.40
		50	2.16	1.26	1.00
12	50	5	1.81	1.51	1.43
		50	3.08	2.80	2.54
	200	5	2.41	1.51	1.43
		50	3.68	2.80	2.54

Figure 6: The start-up delay as a function of segment duration S, one-way Internet delay ID and one-way downstream CDN delay DD; for B = 5Mbps and P = 36s

In summary, results showed that increasing the HAS segment duration can help in overcoming the reduced video quality that occurs when using simple Inter-CDN routing policies, such as UpstreamRR. Nevertheless, long segment durations also have significant disadvantages, such as slower convergence to the optimal quality, an increased start-up delay and greater session lag in live scenarios. This makes them unsuitable in many use-cases, such as live or channel-switching intensive services.

4. Conclusion

In this document, we proposed, evaluated and compared several policies for routing requests and retrieving HAS content chunks distributed across multiple interconnected CDNs. Concretely, the traditional policy, herein called UpstreamRR, in which the original CDN's request router dynamically redirects the end-users towards the CDN currently hosting the requested content, is compared to two novel policies, called DirectRR and DirectCS. These novel policies employ HAS Manifest File rewriting to directly point end-users to the correct CDN (DirectRR) or even the correct content server (DirectCS).

A thorough evaluation, using an open source implementation of the Microsoft Smooth Streaming client algorithm and based on NS-3 simulation results, was conducted. It shows that the end-user QoE suffers greatly as a consequence of the HTTP redirects that occur when employing the standard UpstreamRR policy. Specifically, it was shown that when downloading segments from the downstream CDN, DirectRR and DirectCS result in a much lower buffer starvation rate and start-up delay, as well as an increased video quality compared to UpstreamRR. Additionally, DirectCS significantly outperforms the other two strategies in terms of buffer starvation rate and start-up delay for segments downloaded from the upstream CDN. Finally, the evaluation showed that increasing the segment duration can negate the negative effects of redirects on video quality when using the UpstreamRR policy. However, it also leads to increased start-up delay and slower convergence to the optimal quality.

In summary, these results prove the need for advanced request routing mechanisms, as well as extensive cooperation between interconnected CDNs, to be able to satisfy end-user quality requirements of state-of-the-art HAS-based services. Additionally, the results show the merits of the more complex DirectCS policy compared to the easier to implement DirectRR.

5. Security Considerations

Not applicable.

6. References

6.1. Normative References

[I-D.brandenburg-cdni-has]
Brandenburg, R., Deventer, O., Faucheur, F., and K. Leung, "Models for adaptive-streaming-aware CDN Interconnection", draft-brandenburg-cdni-has-03 (work in progress), July 2012.

6.2. Informative References

[msscode] "Microsoft's SmoothStreaming rate adaptation algorithm v1 source code", <https://slextensions.svn.codeplex.com/svn/trunk/SLExtensions/AdaptiveStreaming/>.

[ns3] "NS-3 discrete event network simulator", <http://www.nsnam.org/>.

[nsc] "Network Simulation Cradle",
<http://research.wand.net.nz/software/nsc.php>.

Authors' Addresses

Jeroen Famaey
Ghent University - iMinds
Gaston Crommenlaan 8/201
Ghent 9050
Belgium

Phone: +32 9 331 49 38
Email: jeroen.famaey@intec.ugent.be

Steven Latre
Ghent University - iMinds
Gaston Crommenlaan 8/201
Ghent 9050
Belgium

Phone: +32 9 331 49 88
Email: steven.latre@intec.ugent.be