Getting Started with Low Latency HLS

Draft Implementation Guide

Version 0.5

May 8, 2019

Introduction

In order to enable low latency video streaming that is scalable to hundreds of thousands of concurrent viewers, Apple is augmenting the HLS protocol with several new features. Accompanying these is a new low-latency mode in Apple HLS clients that can support video latencies of two seconds or less over public networks and which can be easily configured for longer latencies.

Backend production tools and content delivery systems must adopt new rules and mechanisms in order to generate streams that clients will play at low latency. This document outlines what is necessary to produce and deliver such streams.

Overview

How Low Latency HLS Works

Low Latency HLS is an extension of the existing HLS protocol documented in RFC 8216 and draft-pantos-hls-rfc8216bis. There are four new areas of functionality: generation of partial media segments, playlist delta updates, blocking playlist reload, and rendition reports. In addition there is a Server Configuration Profile that distribution systems must support in order to engage the low-latency playback mode.

Signaling Low Latency HLS

A server indicates compatibility with the low latency mode with a new Media Playlist tag, EXT-X-SERVER-CONTROL.

Generation of Partial Segments

Low Latency HLS works by providing a parallel channel for distributing the media at the live edge of the Media Playlist wherein the media is divided into a larger number of smaller files, such as CMAF Chunks. These smaller files are called HLS Partial Segments. Because each Partial Segment has a short duration it can be packaged, published and added to the Media Playlist much earlier than its parent Segment. While regular Segments might be 6s each, an example Partial Segment might be only 200ms. So a first Partial Segment might be published only 200ms after the previous Segment has been published, followed by 29 of its peers, followed at last by a regular 6s Segment which contains the same media as the concatenation of its 30 Partial Segments. In order to reduce Playlist bloat, Partial Segments are removed from the Media Playlist once they are greater (older) than 3 target durations from the live edge.

Partial Segments are added to the Playlist using the new EXT-X-PART tag. Other Media Segment Tags (such as EXT-X-DISCONTINUITY) may only appear at Segment boundaries, not between Partial Segments.

Playlist Delta Updates

Playlists are transferred more frequently with Low Latency HLS. Clients can request and servers can provide Playlist Delta Updates to reduce the transfer cost. These updates replace a considerable portion of the Playlist that the client already has with the new EXT-X-SKIP tag.

Blocking Playlist Reload

In order to support efficient client notification of new Segments and Partial Segments, Low Latency HLS introduces the notion of a blocking Playlist reload request. When a client issues an HTTP GET to request a Playlist update, it can add query parameters to specify that it wants the Playlist response to contain a future segment. It is the responsibility of the server to hold onto the request (block) until a version of the Playlist that contains that segment is available. The client can also indicate that it wishes the server to push the indicated segment to it along with the Playlist response. This eliminates polling and request round trips.

Rendition Reports

When playing at low latency it is essential that the client is able to switch Renditions with a minimum number of round trips in order to perform bit rate adaptation. To support this, the client can ask that a Media Playlist update contains Rendition Reports on different Renditions in the Master Playlist. The Rendition Report is carried in the EXT-X-RENDITION-REPORT tag and carries information such as the last Media Sequence Number and Part currently in the Media Playlist of that Rendition.

Server Configuration Profile

In order to support timely delivery of media, Low Latency HLS requires certain transport features above and beyond what is necessary for regular HLS. Clients must to verify that these features are in place before engaging low latency mode. Because the Low Latency HLS syntax is backward-compatible with existing HLS, clients will fall back to regular-latency HLS playback if they discover that the server does not support any aspect of the required configuration.

Protocol Additions

Playlist Request Query Parameters

The **_HLS_msn**=<N> parameter indicates that the server must hold the request until a playlist contains a Media Segment with media sequence N or later. The server must deliver the entire playlist, even if the requested Media Segment is not the last in the playlist, or in fact if it is no longer in the playlist.

The **_HLS_part**=<M> parameter in combination with _HLS_msn indicates that the server must hold the request until a playlist contains Partial Segment M of media sequence N or later. The first Partial Segment of a segment is _HLS_part=0, the second is _HLS_part=1, etc. The _HLS_-part parameter requires an _HLS_msn parameter.

If the client asks for a Partial Segment number greater than the final part of a Segment, the server must deliver a playlist containing the first Partial Segment of the following Segment.

If a client supplies an _HLS_msn parameter greater than the Media Sequence Number of the last Segment in the Playlist plus one or if it supplies an _HLS_part parameter greater than the last partial segment plus 1 then the server must return 400.

There is a 3 * target duration timeout for blocking requests, after which the server must return 503.

The **_HLS_push**=<0/1> parameter indicates whether the server should push the awaited (Partial) Segment along with the playlist response. 1 means push, 0 means don't push. The absence of an _HLS_push parameter is equivalent to _HLS_push=0.

_HLS_msn and _HLS_part do not trigger blocking if the playlist contains an EXT-X-ENDLIST tag.

The **_HLS_report**=<path> parameter requests that the playlist response contains an EXT-X-RENDITION-REPORT tag for the specified rendition. The path points to the Media Playlist of the specified Rendition. The path may be either relative to the URI of the Media Playlist request being made, or an absolute path (starting with /) on the same server. Multiple report parameters are allowed for different paths.

The **_HLS_skip**=YES parameter is used to request a Playlist Delta Update, in which the earlier portion of the Playlist is replaced with an EXT-X-SKIP tag. The server must ignore this parameter if the Playlist contains an EXT-X-ENDLIST tag.

Clients must sort all Low Latency HLS query parameters in UTF-8 order and place them after any other query parameters in the URL. This increases CDN cache utilization.

New Media Playlist tags

EXT-X-SERVER-CONTROL:<attribute-list>

EXT-X-SERVER-CONTROL allows the server to indicate support for features such as Blocking Playlist Reload and Playlist Delta Updates. Low Latency HLS requires this tag.

All Media Playlists must carry this tag with the same attributes and values.

The **CAN-BLOCK-RELOAD**=YES attribute indicates that the server supports the processing of the _HLS_msn, _HLS_part, _HLS_push and _HLS_report parameters. It is mandatory for Low Latency HLS.

The **CAN-SKIP-UNTIL**=<seconds> attribute indicates that the server can produce Playlist Delta Updates in response to the _HLS_skip=YES parameter. Its value is the Skip Boundary, as a floating-point number of seconds. Segments and their associated tags that are further from the live edge than the Skip Boundary can be replaced by an EXT-X-SKIP tag in a Playlist Delta Update. The Skip Boundary must be at least 6 times the EXT-X-TARGETDURATION.

The **HOLD-BACK**=<seconds> attribute (optional) indicates the server-recommended minimum distance from the live edge at which clients should begin to play or to which they should seek when NOT playing in Low Latency mode (i.e. conventional HLS). Its value is a floating-point number of seconds and must be at least 3 times the EXT-X-TARGETDURATION.

The **PART-HOLD-BACK**=<seconds> attribute (mandatory) indicates the server-recommended hold-back time when playing in Low Latency mode.

PART-HOLD-BACK must be at least PART-TARGET. We recommend that it be at least 3 * PART-TARGET

EXT-X-PART-INF:<attribute-list>

EXT-X-PART-INF is a Media Playlist Tag that provides information about HLS Partial Segments in the Playlist. It is required if a Playlist contains one or more EXT-X-PART tags.

The **PART-TARGET**=<s> attribute (mandatory) indicates the part target duration in floating-point seconds and is the maximum duration of any Partial Segment.

All Partial Segments except the last part of a Segment must have a duration of at least 85% of PART-TARGET

EXT-X-PART:<attribute-list>

EXT-X-PART identifies a Partial Segment in the playlist.

The set of EXT-X-PART tags between EXTINF tags must represent the same set of media as the Media Segment of the following EXTINF tag. The Media Segment containing the same media as a set of Partial Segments is called the Parent Segment of those Partial Segments.

All Media Segment Tags with the exception of EXT-X-DATERANGE, EXT-X-BYTERANGE and EXT-X-GAP that are applied to a Parent Segment must appear before the first EXT-X-PART tag of the Parent Segment,. This includes EXT-X-MAP, EXT-X-DISCONTINUITY and EXT-X-KEY.

Remove EXT-X-PART tags from the playlist after they are greater than 3 target durations from the end of the playlist. Partial Segments are primarily useful for navigating close to the live edge, after that their presence does not justify the increase in the playlist size and responsibility of retaining the parallel Partial Segment stream on the server.

The **DURATION**=<s> attribute (mandatory) indicates the duration of the part in floating-point seconds.

The **URI**=<url> attribute (mandatory) indicates the URI for the part.

The **INDEPENDENT**=YES attribute indicates that the Partial Segment contains (at least) the start of an independent frame; mandatory if applicable.

The **BYTERANGE**=<n>[@<o>] attribute indicates that the Partial Segment is a subrange of the URI.

The **GAP**=YES attribute indicates that the partial segment is not available. The rules that apply to a Media Segment with an EXT-X-GAP tag applied to it also apply to Partial Segments with a GAP=YES attribute. A Parent Segment must have an EXT-X-GAP tag applied to it if one or more of its Partial Segments have a GAP=YES attribute. The EXT-X-GAP tag should appear immediately after the first EXT-X-PART tag in the Parent Segment with a GAP=YES attribute.

EXT-X-RENDITION-REPORT:<attribute-list>

EXT-X-RENDITION-REPORT carries information about an associated rendition that is as up-to-date as the Playlist that contains it. It is added to the Playlist in response to a "report" query parameter

A server may return more EXT-RENDITION-REPORT tags than requested.

The **URI**=<uri> attribute (mandatory) indicates the Media Playlist described in the report. It should be the same string as the report query parameter.

The **LAST-MSN**=<N> attribute (mandatory) indicates the media sequence number of the last segment currently in the specified rendition.

The **LAST-PART**=<M> attribute indicates the last part of the segment specified by the media sequence number currently in the specified rendition.

Currently the prototype tools also generate LAST-I-MSN and LAST-I-PART parameters to indicate the MSN and part of the last independent part. But these may not make it into the spec as they are currently unused.

EXT-X-SKIP:<attribute-list>

When a server issues a Playlist Delta Update it replaces Segments earlier than the Skip Boundary and their associated tags with an **EXT-X-SKIP** tag.

The EXT-X-SKIP tag replaces Segment URI lines and all of the following tags that are applied to those Segments: EXTINF, EXT-X-BYTERANGE, EXT-X-DISCONTINUITY, EXT-X-KEY, EXT-X-MAP, EXT-X-PROGRAM-DATE-TIME, EXT-X-GAP, and EXT-X-BITRATE. All other tags must remain in the Playlist Delta Update.

The **SKIPPED-SEGMENTS**=<N> attribute (mandatory) indicates how many Media Segments and associated tags were replaced by the EXT-X-SKIP tag.

If a client receive a Playlist containing an EXT-X-SKIP tag and discovers that it does not already have all of the information that was skipped, it must obtain a complete copy of the Playlist by reissuing its Playlist request without the _HLS_skip=YES parameter.

A Playlist containing an EXT-X-SKIP tag must have an EXT-X-VERSION tag with a value of 9 or higher.

Server Configuration Profile

New Rules

You must serve Low Latency HLS streams via HTTP/2 as it requires several of its features: multi-stream control, H2 Push, and H2 Ping.

Servers must support H2 priority control (dependencies and weights).

We recommend TCP. We are not committing to supporting QUIC in the first release.

Each server must offer the entire set of tiers in the master playlist. This allows rapid tier switching without connection reestablishment.

TCP implementations must support Selective Acknowledgment (SACK) across the entire route from client to server. We also recommend setting Explicit Congestion Notification (ECN) during congestion, and using TCP timestamps, TAIL LOSS probe and TCP RACK. These additions improve the performance of TCP loss recovery. See RFC 2018, RFC 3168, RFC 7323, and IETF draft-ietf-tcpm-rack for more information about these TCP options.

Playlists (but not segments) must be gzip'd. This speeds up Media Playlist reload and Rendition switching.

CDNs and proxy caches must recognize client playlist requests that match an existing request that is currently outstanding to the origin and hold the duplicate requests until the origin responds to the existing request. This is more critical in Low Latency HLS due to its requirements for an active origin. CDNs have different names for this feature; for example, Apache Traffic Server calls it reader-while-writer.

Low Latency HLS allows longer caching of playlists without detriment to clients. We recommend caching successful playlist request/responses with Low Latency HLS query-parameters for 6 target durations. We recommend caching unsuccessful playlist request/responses with Low Latency HLS query-parameters (such as 404s) for 1 target duration. (We recommend caching successful unadorned playlist request/responses for 0.5 target duration and unsuccessful request/responses for 1 target duration.)

Origins should use cache-control headers to indicate the desired cache lifetime.

Different renditions must update in sync, to within 1 part-target duration.

Tweaks to existing media authoring rules for Low Latency HLS streams:

Media Playlists must have EXT-PROGRAM-DATE-TIME tags. This allows more-precise mapping between segments across renditions. (Note that real-time clocks are NOT required to be synchronization between client and server).

We continue to recommend a 6s target duration.

We recommend GOP sizes of 1-2s. Smaller GOPs support faster switching between renditions.

Example of a Low Latency HLS playlist

```
#EXTM3U
# This playlist was a response to: GET https://example.com/2M/waitForMSN.php?_HLS_msn=273&_HLS_part=2&_HLS_report=../1M/waitForMSN.php&_HLS_report=../4M/waitForMSN.php
#EXT-X-TARGETDURATION:4
#EXT-X-VERSION:3
#EXT-X-BLOCKING-RELOAD:PART-TARGET=0.33334,HOLD-BACK=1.0
#EXT-X-MEDIA-SEQUENCE:268
#EXT-X-PROGRAM-DATE-TIME:2019-02-14T02:13:36.106Z
#EXTINF:4.00008,
fileSequence268.ts
#EXTINF:4.00008,
fileSequence269.ts
#EXTINF:4.00008,
fileSequence270.ts
#EXT-X-PART:DURATION=0.33334,URI="filePart271.1.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.2.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.3.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.4.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.5.ts",INDEPENDENT=YES
#EXT-X-PART:DURATION=0.33334,URI="filePart271.6.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.7.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.8.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.9.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.10.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart271.11.ts",INDEPENDENT=YES
#EXT-X-PART:DURATION=0.33334,URI="filePart271.12.ts"
#EXTINF:4.00008,
fileSequence271.ts
#EXT-X-PART:DURATION=0.33334,URI="filePart272.a.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.b.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.c.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.d.ts"
```

```
#EXT-X-PART:DURATION=0.33334,URI="filePart272.e.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.f.ts",INDEPENDENT=YES
#EXT-X-PART:DURATION=0.33334,URI="filePart272.g.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.h.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.i.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.j.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.k.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart272.l.ts"
#EXTINF:4.00008,
fileSequence272.ts
#EXT-X-PART:DURATION=0.33334,URI="filePart273.1.ts",INDEPENDENT=YES
#EXT-X-PART:DURATION=0.33334,URI="filePart273.2.ts"
#EXT-X-PART:DURATION=0.33334,URI="filePart273.3.ts"

#EXT-X-RENDITION-REPORT:URI="../1M/waitForMSN.php",LAST-MSN=273,LAST-PART=2
#EXT-X-RENDITION-REPORT:URI="../4M/waitForMSN.php",LAST-MSN=273,LAST-PART=1
```

Document Revision History

This table describes the changes to *Getting Started with Low Latency HLS*

Date	Revision	Notes
2019-02-20	0.1	First revision
2019-03-04	0.2	Add GAP attribute. Clarify position of Media Segment Tags. Add EXT-X-GAP & EXT-X-BYTERANGE to list of exceptions.
2019-03-31	0.3	Tweak _HLS_report to allow absolute paths. Add negative caching guidance.
2019-04-09	0.4	Clarify that server delivers entire playlist.
2019-05-07	0.5	Add delta playlist updates. Replace BLOCKING-RELOAD with SERVER-CONTROL and PART-INF.