Informational
Internet-Draft
Obsoletes: 8216 (if approved)
Intended status: Informational
Expires: October 29, 2021

R. Pantos, Ed.
Apple Inc.

April 27, 2021

HTTP Live Streaming 2nd Edition
draft-pantos-hls-rfc8216bis-09

Abstract

This document obsoletes RFC 8216. It describes a protocol for transferring unbounded streams of multimedia data. It specifies the data format of the files and the actions to be taken by the server (sender) and the clients (receivers) of the streams. It describes version 10 of this protocol.

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 https://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 October 29, 2021.

Copyright Notice

Table of Contents

1. Introduction to HTTP Live Streaming 5
2. Overview 5
3. Media Segments 7
3.1. Supported Media Segment Formats 7
3.1.1. MPEG-2 Transport Streams 8
3.1.2. Fragmented MPEG-4 8
3.1.3. Packed Audio 9
3.1.4. WebVTT 9
3.1.5. IMSC Subtitles 10
3.2. Partial Segments 11
4. Playlists 11
4.1. Definition of a Playlist 12
4.2. Attribute Lists 13
4.3. Variable Substitution 14
4.4. Playlist Tags 15
4.4.1. Basic Tags 15
4.4.1.1. EXTM3U 15
4.4.1.2. EXT-X-VERSION 15
4.4.2. Media or Master Playlist Tags 15
4.4.2.1. EXT-X-INDEPENDENT-SEGMENTS 16
4.4.2.2. EXT-X-START 16
4.4.2.3. EXT-X-DEFINE 17
4.4.3. Media Playlist Tags 18
4.4.3.1. EXT-X-TARGETDURATION 18
4.4.3.2. EXT-X-MEDIA-SEQUENCE 18
4.4.3.3. EXT-X-DISCONTINUITY-SEQUENCE 19
4.4.3.4. EXT-X-ENDLIST 19
4.4.3.5. EXT-X-PLAYLIST-TYPE 20
4.4.3.6. EXT-X-I-FRAMES-ONLY 20
4.4.3.7. EXT-X-PART-INF 21
4.4.3.8. EXT-X-SERVER-CONTROL 21
4.4.4. Media Segment Tags 22
4.4.4.1. EXTINF 22
4.4.4.2. EXT-X-BYTERANGE 23
4.4.4.3. EXT-X-DISCONTINUITY 23
4.4.4.4. EXT-X-KEY 24
4.4.4.5. EXT-X-MAP 26

4.4.4.6. EXT-X-PROGRAM-DATE-TIME 27
4.4.4.7. EXT-X-GAP 27
4.4.4.8. EXT-X-BITRATE 28
4.4.4.9. EXT-X-PART 28
4.4.5. Media Metadata Tags 29
4.4.5.1. EXT-X-DATERANGE 29
4.4.5.1.1. Mapping SCTE-35 into EXT-X-DATERANGE 31
4.4.5.2. EXT-X-SKIP 33
4.4.5.3. EXT-X-PRELOAD-HINT 34
4.4.5.4. EXT-X-RENDITION-REPORT 35
4.4.6. Master Playlist Tags 36
4.4.6.1. EXT-X-MEDIA 36
4.4.6.1.1. Rendition Groups 39
4.4.6.2. EXT-X-STREAM-INF 40
4.4.6.2.1. Alternative Renditions 46
4.4.6.3. EXT-X-I-FRAME-STREAM-INF 46
4.4.6.4. EXT-X-SESSION-DATA 47
4.4.6.5. EXT-X-SESSION-KEY 48
5. Key Files 48
5.1. Structure of Key Files 48
5.2. IV for AES-128 49
6. Client/Server Responsibilities 49
6.1. Introduction 49
6.2. Server Responsibilities 49
6.2.1. General Server Responsibilities 49
6.2.2. Live Playlists 52
6.2.3. Encrypting Media Segments 54
6.2.4. Providing Variant Streams 54
6.2.5. Delivery Directives Interface 56
6.2.5.1. Playlist Delta Updates 56
6.2.5.2. Blocking Playlist Reload 57
6.2.6. Providing Preload Hints 58
6.3. Client Responsibilities 59
6.3.1. General Client Responsibilities 59
6.3.2. Loading the Media Playlist File 60
6.3.3. Playing the Media Playlist File 61
6.3.4. Reloading the Media Playlist File 62
6.3.5. Determining the Next Segment to Load 63
6.3.6. Decrypting Encrypted Media Segments 63
6.3.7. Requesting Playlist Delta Updates 64
6.3.8. Issuing Blocking Requests 65
7. Protocol Version Compatibility 66
8. Playlist Examples 68
8.1. Simple Media Playlist 68
8.2. Live Media Playlist Using HTTPS 68
8.3. Playlist with Encrypted Media Segments 68
8.4. Master Playlist 69
8.5. Master Playlist with I-Frames 69

8.6. Master Playlist with Alternative Audio 70
8.7. Master Playlist with Alternative Video 70
8.8. Session Data in a Master Playlist 71
8.9. CHARACTERISTICS Attribute Containing Multiple Characteristics 71
8.10. EXT-X-DATERANGE Carrying SCTE-35 Tags 72
8.11. Low-Latency Playlist 72
9. Contributors 73
10. IANA Considerations 73
11. Security Considerations 75
12. References 75
12.1. Normative References 75
12.2. Informative References 79
Appendix A. Changes from RFC 8216 80
Appendix B. Server Configuration Profiles 81
B.1. Low-Latency Server Configuration Profile 81
Appendix C. Low-Latency CDN Tune-in 83
Author's Address 84

1. Introduction to HTTP Live Streaming

HTTP Live Streaming provides a reliable, cost-effective means of delivering continuous and long-form video over the Internet. It allows a receiver to adapt the bit rate of the media to the current network conditions in order to maintain uninterrupted playback at the best possible quality. It supports interstitial content boundaries. It provides a flexible framework for media encryption. It can efficiently offer multiple renditions of the same content, such as audio translations. It offers compatibility with large-scale HTTP caching infrastructure to support delivery to large audiences. It supports low-latency playback at scale.

Since its first draft publication in 2009, HTTP Live Streaming has been implemented and deployed by a wide array of content producers, tools vendors, distributors, and device manufacturers. In the subsequent ten years the protocol has been refined by extensive review and discussion with a variety of media streaming implementors.

The purpose of this document is to facilitate interoperability between HTTP Live Streaming implementations by describing the media transmission protocol. Using this protocol, a client can receive a continuous stream of media from a server for concurrent presentation.

This document describes version 10 of the protocol.

2. Overview

A multimedia presentation is specified by a Uniform Resource Identifier (URI) [RFC3986] to a Playlist.

A Playlist is either a Media Playlist or a Master Playlist. Both are UTF-8 text files containing URIs and descriptive tags.

A Media Playlist contains a list of Media Segments, which, when played sequentially, will play the multimedia presentation.

Here is an example of a Media Playlist:

```
#EXTM3U
#EXT-X-TARGETDURATION:10

#EXTINF:9.009,
http://media.example.com/first.ts
#EXTINF:9.009,
http://media.example.com/second.ts
#EXTINF:3.003,
http://media.example.com/third.ts
```

The first line is the format identifier tag #EXTM3U. The line containing #EXT-X-TARGETDURATION says that all Media Segments will be 10 seconds long or less. Then, three Media Segments are declared. The first and second are 9.009 seconds long; the third is 3.003 seconds.

To play this Playlist, the client first downloads it and then downloads and plays each Media Segment declared within it. The client reloads the Playlist as described in this document to discover any added segments. Data SHOULD be carried over HTTP [RFC7230], but, in general, a URI can specify any protocol that can reliably transfer the specified resource on demand.

A more complex presentation can be described by a Master Playlist. A Master Playlist provides a set of Variant Streams, each of which describes a different version of the same content.

A Variant Stream includes a Media Playlist that specifies media encoded at a particular bit rate, in a particular format, and at a particular resolution for media containing video.

A Variant Stream can also specify a set of Renditions. Renditions are alternate versions of the content, such as audio produced in different languages or video recorded from different camera angles.

Clients should switch between different Variant Streams to adapt to network conditions. Clients should choose Renditions based on user preferences.

Certain streams can be played in Low-Latency Mode. Low-Latency Mode refers to the combined use of Partial Segments, Blocking Playlist Reload and preload hinting to enable playback at a reduced delay from live.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Media Segments

A Media Playlist contains a series of Media Segments that make up the overall presentation. A Media Segment is specified by a URI and optionally a byte range.

The duration of each Media Segment is indicated in the Media Playlist by its EXTINF tag (Section 4.4.4.1).

Each segment in a Media Playlist has a unique integer Media Sequence Number. The Media Sequence Number of the first segment in the Media Playlist is either 0 or declared in the Playlist (Section 4.4.3.2). The Media Sequence Number of every other segment is equal to the Media Sequence Number of the segment that precedes it plus one.

Each Media Segment MUST carry the continuation of the encoded bitstream from the end of the segment with the previous Media Sequence Number, where values in a series such as timestamps and Continuity Counters MUST continue uninterrupted. The only exceptions are the first Media Segment ever to appear in a Media Playlist and Media Segments that are explicitly signaled as discontinuities (Section 4.4.4.3). Unmarked media discontinuities can trigger playback errors.

Any Media Segment that contains video SHOULD include enough information to initialize a video decoder and decode a continuous set of frames that includes the final frame in the Segment; network efficiency is optimized if there is enough information in the Segment to decode all frames in the Segment. For example, any Media Segment containing H.264 video SHOULD contain an Instantaneous Decoding Refresh (IDR); frames prior to the first IDR will be downloaded but possibly discarded.

3.1. Supported Media Segment Formats

All Media Segments MUST be in a format described in this section. Transport of other media file formats is not defined.

Some media formats require a common sequence of bytes to initialize a parser before a Media Segment can be parsed. This format-specific sequence is called the Media Initialization Section. The Media Initialization Section can be specified by an EXT-X-MAP tag (Section 4.4.4.5). The Media Initialization Section MUST NOT contain sample data.

3.1.1. MPEG-2 Transport Streams

MPEG-2 Transport Streams are specified by [ISO_13818].

The Media Initialization Section of an MPEG-2 Transport Stream Segment is a Program Association Table (PAT) followed by a Program Map Table (PMT).

Transport Stream Segments MUST contain a single MPEG-2 Program; playback of Multi-Program Transport Streams is not defined. Each Transport Stream Segment MUST contain a PAT and a PMT, or have an EXT-X-MAP tag (Section 4.4.4.5) applied to it. The first two Transport Stream packets in a Segment without an EXT-X-MAP tag SHOULD be a PAT and a PMT.

3.1.2. Fragmented MPEG-4

MPEG-4 Fragments are specified by the ISO Base Media File Format [ISOBMFF]. Unlike regular MPEG-4 files that have a Movie Box ('moov') that contains sample tables and a Media Data Box ('mdat') containing the corresponding samples, an MPEG-4 Fragment consists of a Movie Fragment Box ('moof') containing a subset of the sample table and a Media Data Box containing those samples. Use of MPEG-4 Fragments does require a Movie Box for initialization, but that Movie Box contains only non-sample-specific information such as track and sample descriptions.

A Fragmented MPEG-4 (fMP4) Segment is a "segment" as defined by Section 3 of [ISOBMFF], including the constraints on Media Data Boxes in Section 8.16 of [ISOBMFF].

The Media Initialization Section for an fMP4 Segment is an ISO Base Media File that can initialize a parser for that Segment.

Broadly speaking, fMP4 Segments and Media Initialization Sections are [ISOBMFF] files that also satisfy the constraints described in this section.

The Media Initialization Section for an fMP4 Segment MUST contain a File Type Box ('ftyp') containing a brand that is compatible with 'iso6' or higher. The File Type Box MUST be followed by a Movie Box. The Movie Box MUST contain a Track Box ('trak') for every Track Fragment Box ('traf') in the fMP4 Segment, with matching track_ID. Each Track Box SHOULD contain a sample table, but its sample count MUST be zero. Movie Header Boxes ('mvhd') and Track Header Boxes ('tkhd') MUST have durations of zero. The Movie Box MUST contain a Movie Extends Box ('mvex'); it SHOULD follow the last Track Box.

Note that a Common Media Application Format [CMAF] Header meets all these requirements.

In an fMP4 Segment, every Track Fragment Box MUST contain a Track Fragment Decode Time Box ('tfdt'). fMP4 Segments MUST use movie-fragment-relative addressing. fMP4 Segments MUST NOT use external data references. Note that a CMAF Segment meets these requirements.

An fMP4 Segment in a Playlist containing the EXT-X-I-FRAMES-ONLY tag (Section 4.4.3.6) MAY omit the portion of the Media Data Box following the intra-coded frame (I-frame) sample data.

This specification makes no additional restrictions on [ISOBMFF] boxes or box order. However, fMP4 Segments that indicate compatibility with an additional standard, such as [CMAF], SHOULD comply with whatever rules that standard requires.

Each fMP4 Segment in a Media Playlist MUST have an EXT-X-MAP tag applied to it.

3.1.3. Packed Audio

A Packed Audio Segment contains encoded audio samples and ID3 tags that are simply packed together with minimal framing and no per-sample timestamps. Supported Packed Audio formats are Advanced Audio Coding (AAC) with Audio Data Transport Stream (ADTS) framing [ISO_13818_7], MP3 [ISO_13818_3], AC-3 [AC_3], and Enhanced AC-3 [AC_3].

A Packed Audio Segment has no Media Initialization Section.

Each Packed Audio Segment MUST signal the timestamp of its first sample with an ID3 Private frame (PRIV) tag [ID3] at the beginning of the segment. The ID3 PRIV owner identifier MUST be "com.apple.streaming.transportStreamTimestamp". The ID3 payload MUST be a 33-bit MPEG-2 Program Elementary Stream timestamp expressed as a big-endian eight-octet number, with the upper 31 bits set to zero. Clients SHOULD NOT play Packed Audio Segments without this ID3 tag.

3.1.4. WebVTT

A WebVTT Segment is a section of a WebVTT [WebVTT] file. WebVTT Segments carry subtitles.

The Media Initialization Section of a WebVTT Segment is the WebVTT header.

Each WebVTT Segment MUST contain all subtitle cues that are intended to be displayed during the period indicated by the segment EXTINF duration. The start time offset and end time offset of each cue MUST indicate the total display time for that cue, even if part of the cue time range is outside the Segment period. A WebVTT Segment MAY contain no cues; this indicates that no subtitles are to be displayed during that period.

Under certain conditions, like live streaming, where it is not possible to know the cue duration at the time of the segment creation and the subtitle cue interval is split over multiple Segments, the cue time range in each Segment MAY be limited to the WebVTT time range covered by the Segment.

Each WebVTT Segment MUST either start with a WebVTT header or have an EXT-X-MAP tag applied to it.

In order to synchronize timestamps between audio/video and subtitles, an X-TIMESTAMP-MAP metadata header SHOULD be added to each WebVTT header. This header maps WebVTT cue timestamps to media timestamps in other Renditions of the Variant Stream. Its format is:

X-TIMESTAMP-MAP=LOCAL:<cue time>,MPEGTS:<media time>
e.g., X-TIMESTAMP-MAP=LOCAL:00:00:00.000,MPEGTS:900000

indicating the media time to which the cue time MUST be mapped. The cue timestamp in the LOCAL attribute MAY fall outside the range of time covered by the segment.

The MPEGTS media timestamp MUST use a 90KHz timescale, even when non-WebVTT Media Segments use a different timescale.

If a WebVTT segment does not have the X-TIMESTAMP-MAP, the client MUST assume that the WebVTT cue time of 0 maps to an media timestamp of 0.

When synchronizing WebVTT with PES timestamps, clients SHOULD account for cases where the 33-bit PES timestamps have wrapped and the WebVTT cue times have not. When the PES timestamp wraps, the WebVTT segment SHOULD have a X-TIMESTAMP-MAP header that maps the current WebVTT time to the new (low valued) PES timestamp.

3.1.5. IMSC Subtitles

An IMSC Segment is a Fragmented MPEG-4 (Section 3.1.2) Media Segment that carries subtitle media according to MPEG-4 Part 30 [MP4_TIMED_TEXT]. This subtitle media MUST comply with the Text Profile of IMSC1 [IMSC1].

The Media Initialization Section of an IMSC Segment is specified in Section 3.1.2.

Each IMSC Segment MUST contain all subtitle samples that are intended to be displayed during the period indicated by the segment EXTINF duration. Each Segment MUST contain definitions for all styles which are applied to any part of any sample in the Segment.

3.2. Partial Segments

One component of viewer delay in a live stream is publishing latency: a Segment cannot be distributed until it has been completely encoded and packaged. A long Segment encoded in real-time introduces a delay equal to its duration. Partial Segments provide a parallel channel for distributing media at the live edge of the Media Playlist, where the media is divided into a larger number of smaller pieces, such as CMAF Chunks. These subsets are called 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.

A Partial Segment MUST be in one of the Supported Media Segment Formats described in Section 3.1. A Partial Segment is associated with a regular Media Segment, called its Parent Segment, by appearing before it in the Media Playlist, and after the previous Media Segment. Partial Segments are identified by the EXT-X-PART tag (Section 4.4.4.9).

A Partial Segment MUST contain a subset of the media samples in its Parent Segment. A Parent Segment and its entire set of Partial Segments MUST contain the same set of media samples, with the same timing and metadata.

Each Partial Segment has a Part Index, which is an integer indicating the position of the Partial Segment within its Parent Segment. The first Partial Segment has a Part Index of zero.

Each Partial Segment also has a Media Sequence Number, which is equal to the Media Sequence Number of its Parent Segment.

4. Playlists

This section describes the Playlist files used by HTTP Live Streaming. In this section, "MUST" and "MUST NOT" specify the rules for the syntax and structure of legal Playlist files. Playlists that violate these rules are invalid; clients MUST fail to parse them. See Section 6.3.2.

The format of the Playlist files is derived from the M3U [M3U] playlist file format and inherits two tags from that earlier file format: EXTM3U (Section 4.4.1.1) and EXTINF (Section 4.4.4.1).

In the specification of tag syntax, a string enclosed by <> identifies a tag parameter; its specific format is described in its tag definition. If a parameter is further surrounded by [], it is optional; otherwise, it is required.

Each Playlist file MUST be identifiable either by the path component of its URI or by HTTP Content-Type. In the first case, the path MUST end with either .m3u8 or .m3u. In the second, the HTTP Content-Type MUST be "application/vnd.apple.mpegurl" or "audio/mpegurl". Clients SHOULD refuse to parse Playlists that are not so identified.

4.1. Definition of a Playlist

Playlist files MUST be encoded in UTF-8 [RFC3629]. They MUST NOT contain any Byte Order Mark (BOM); clients SHOULD fail to parse Playlists that contain a BOM or do not parse as UTF-8. Playlist files MUST NOT contain UTF-8 control characters (U+0000 to U+001F and U+007F to U+009F), with the exceptions of CR (U+000D) and LF (U+000A). All character sequences MUST be normalized according to Unicode normalization form "NFC" [UNICODE]. Note that US-ASCII [US_ASCII] conforms to these rules.

Lines in a Playlist file are terminated by either a single line feed character or a carriage return character followed by a line feed character. Each line is a URI, is blank, or starts with the character '#'. Blank lines are ignored. Whitespace MUST NOT be present, except for elements in which it is explicitly specified.

Lines that start with the character '#' are either comments or tags. Tags begin with #EXT. They are case sensitive. All other lines that begin with '#' are comments and SHOULD be ignored.

A URI line identifies a Media Segment or a Playlist file (see Section 4.4.6.2). Each Media Segment is specified by a URI and the tags that apply to it.

A Playlist is a Media Playlist if all URI lines in the Playlist identify Media Segments. A Playlist is a Master Playlist if all URI lines in the Playlist identify Media Playlists. A Playlist MUST be either a Media Playlist or a Master Playlist; all other Playlists are invalid.

A URI in a Playlist, whether it is a URI line or part of a tag, MAY be relative. Any relative URI is considered to be relative to the URI of the Playlist that contains it.

The duration of a Media Playlist is the sum of the durations of the Media Segments within it.

The segment bit rate of a Media Segment is the size of the Media Segment divided by its EXTINF duration (Section 4.4.4.1). Note that this includes container overhead but does not include overhead imposed by the delivery system, such as HTTP, TCP, or IP headers.

The peak segment bit rate of a Media Playlist is the largest bit rate of any contiguous set of segments whose total duration is between 0.5 times the Target Duration and 1.5 times the Target Duration plus 0.5 seconds (since media segments may exceed the Target Duration by up to 0.5 seconds). The bit rate of a set is calculated by dividing the sum of the segment sizes by the sum of the segment durations.

The average segment bit rate of a Media Playlist is the sum of the sizes (in bits) of every Media Segment in the Media Playlist, divided by the Media Playlist duration. Note that this includes container overhead, but not HTTP or other overhead imposed by the delivery system.

4.2. Attribute Lists

Certain tags have values that are attribute-lists. An attribute-list is a comma-separated list of attribute/value pairs with no whitespace.

An attribute/value pair has the following syntax:

AttributeName=AttributeValue

An AttributeName is an unquoted string containing characters from the set [A..Z], [0..9], and '-'. Therefore, AttributeNames contain only uppercase letters, not lowercase. There MUST NOT be any whitespace between the AttributeName and the '=' character, nor between the '=' character and the AttributeValue.

An AttributeValue is one of the following:

- decimal-integer: an unquoted string of characters from the set [0..9] expressing an integer in base-10 arithmetic in the range from 0 to 2^64-1 (18446744073709551615). A decimal-integer may be from 1 to 20 characters long.

- hexadecimal-sequence: an unquoted string of characters from the set [0..9] and [A..F] that is prefixed with 0x or 0X. The maximum length of a hexadecimal-sequence depends on its AttributeNames.

- decimal-floating-point: an unquoted string of characters from the set [0..9] and '.' that expresses a non-negative floating-point number in decimal positional notation.

- signed-decimal-floating-point: an unquoted string of characters from the set [0..9], '-', and '.' that expresses a signed floating-point number in decimal positional notation.

- quoted-string: a string of characters within a pair of double quotes (0x22). The following characters MUST NOT appear in a quoted-string: line feed (0xA), carriage return (0xD), or double quote (0x22). The string MUST be non-empty, unless specifically allowed. Quoted-string AttributeValues SHOULD be constructed so that byte-wise comparison is sufficient to test two quoted-string AttributeValues for equality. Note that this implies case-sensitive comparison.

- enumerated-string: an unquoted character string from a set that is explicitly defined by the AttributeName. An enumerated-string will never contain double quotes ("), commas (,), or whitespace.

- decimal-resolution: two decimal-integers separated by the "x" character. The first integer is a horizontal pixel dimension (width); the second is a vertical pixel dimension (height).

The type of the AttributeValue for a given AttributeName is specified by the attribute definition.

A given AttributeName MUST NOT appear more than once in a given attribute-list. Clients SHOULD refuse to parse such Playlists.

4.3. Variable Substitution

The following Playlist elements are subject to variable substitution:

- URI lines

- quoted-string AttributeValues

- hexadecimal-sequence AttributeValues

A Variable Reference is a string of the form "{$" (0x7B,0x24) followed by a Variable Name followed by "}" (0x7D). Variable Names are defined by the EXT-X-DEFINE tag (Section 4.4.2.3).

See Section 6.3.1 for more information about variable substitution.

4.4. Playlist Tags

Playlist tags specify either global parameters of the Playlist or information about the Media Segments or Media Playlists that appear after them.

4.4.1. Basic Tags

These tags are allowed in both Media Playlists and Master Playlists.

4.4.1.1. EXTM3U

The EXTM3U tag indicates that the file is an Extended M3U [M3U] Playlist file. It MUST be the first line of every Media Playlist and every Master Playlist. Its format is:

#EXTM3U

4.4.1.2. EXT-X-VERSION

The EXT-X-VERSION tag indicates the compatibility version of the Playlist file, its associated media, and its server.

The EXT-X-VERSION tag applies to the entire Playlist file. Its format is:

#EXT-X-VERSION:<n>

where n is an integer indicating the protocol compatibility version number.

It MUST appear in all Playlists containing tags or attributes that are not compatible with protocol version 1 to support interoperability with older clients. Section 7 specifies the minimum value of the compatibility version number for any given Playlist file.

A Playlist file MUST NOT contain more than one EXT-X-VERSION tag. If a client encounters a Playlist with multiple EXT-X-VERSION tags, it MUST fail to parse it.

4.4.2. Media or Master Playlist Tags

The tags in this section can appear in either Master Playlists or Media Playlists. If one of these tags appears in a Master Playlist, it SHOULD NOT appear in any Media Playlist referenced by that Master

Playlist. A tag that appears in both MUST have the same value; otherwise, clients SHOULD ignore the value in the Media Playlist(s).

Tags in this section MUST NOT appear more than once in a Playlist. If one does, clients MUST fail to parse the Playlist. The only exception to this rule is EXT-X-DEFINE, which MAY appear more than once.

4.4.2.1. EXT-X-INDEPENDENT-SEGMENTS

The EXT-X-INDEPENDENT-SEGMENTS tag indicates that all media samples in a Media Segment can be decoded without information from other segments. It applies to every Media Segment in the Playlist.

Its format is:

#EXT-X-INDEPENDENT-SEGMENTS

If the EXT-X-INDEPENDENT-SEGMENTS tag appears in a Master Playlist, it applies to every Media Segment in every Media Playlist in the Master Playlist.

4.4.2.2. EXT-X-START

The EXT-X-START tag indicates a preferred point at which to start playing a Playlist. By default, clients SHOULD start playback at this point when beginning a playback session. This tag is OPTIONAL.

Its format is:

#EXT-X-START:<attribute-list>

The following attributes are defined:

TIME-OFFSET

The value of TIME-OFFSET is a signed-decimal-floating-point number of seconds. A positive number indicates a time offset from the beginning of the Playlist. A negative number indicates a negative time offset from the end of the last Media Segment in the Playlist. This attribute is REQUIRED.

The absolute value of TIME-OFFSET SHOULD NOT be larger than the Playlist duration. If the absolute value of TIME-OFFSET exceeds the duration of the Playlist, it indicates either the end of the Playlist (if positive) or the beginning of the Playlist (if negative).

If the Playlist does not contain the EXT-X-ENDLIST tag, the TIME-OFFSET SHOULD NOT be within three Target Durations of the end of the Playlist file.

PRECISE

The value is an enumerated-string; valid strings are YES and NO. If the value is YES, clients SHOULD start playback at the Media Segment containing the TIME-OFFSET, but SHOULD NOT render media samples in that segment whose presentation times are prior to the TIME-OFFSET. If the value is NO, clients SHOULD attempt to render every media sample in that segment. This attribute is OPTIONAL. If it is missing, its value should be treated as NO.

4.4.2.3. EXT-X-DEFINE

The EXT-X-DEFINE tag provides a Playlist variable definition or declaration. This tag is OPTIONAL.

Its format is:

#EXT-X-DEFINE:<attribute-list>

The following attributes are defined:

NAME

The value is a quoted-string which specifies the Variable Name. All characters in the quoted-string MUST be from the following set: [a..z], [A..Z], [0..9], '-', and '_'.

VALUE

The value is a quoted-string which specifies the Variable Value. This attribute is REQUIRED if the EXT-X-DEFINE tag has a NAME attribute. The quoted-string MAY be empty.

IMPORT

The value is a quoted-string which specifies the Variable Name and indicates that its value is that of the variable of the same name in the Master Playlist. EXT-X-DEFINE tags containing the IMPORT attribute MUST NOT occur in Master Playlists; they are only allowed in Media Playlists.

If the IMPORT attribute value does not match any Variable Name declared in the Master Playlist, or if the Media Playlist was not

loaded from a Master Playlist, the parser MUST fail to parse the Playlist.

An EXT-X-DEFINE tag MUST contain either a NAME or an IMPORT attribute, but not both.

An EXT-X-DEFINE tag MUST NOT specify the same Variable Name as any other EXT-X-DEFINE tag in the same Playlist. Parsers that encounter duplicate Variable Name declarations MUST fail to parse the Playlist.

Variable Names are case-sensitive.

EXT-X-DEFINE tags do NOT implicitly persist across Playlist reloads.

4.4.3. Media Playlist Tags

Media Playlist tags describe global parameters of the Media Playlist. There MUST NOT be more than one Media Playlist tag of each type in any Media Playlist.

A Media Playlist tag MUST NOT appear in a Master Playlist

4.4.3.1. EXT-X-TARGETDURATION

The EXT-X-TARGETDURATION tag specifies the Target Duration, an upper bound on the duration of all Media Segments in the Playlist. The EXTINF duration of each Media Segment in a Playlist file, when rounded to the nearest integer, MUST be less than or equal to the Target Duration. Longer segments can trigger playback stalls or other errors. It applies to the entire Playlist file. Its format is:

#EXT-X-TARGETDURATION:<s>

where s is a decimal-integer indicating the Target Duration in seconds. The EXT-X-TARGETDURATION tag is REQUIRED.

4.4.3.2. EXT-X-MEDIA-SEQUENCE

The EXT-X-MEDIA-SEQUENCE tag indicates the Media Sequence Number of the first Media Segment that appears in a Playlist file. Its format is:

#EXT-X-MEDIA-SEQUENCE:<number>

where number is a decimal-integer.

If the Media Playlist file does not contain an EXT-X-MEDIA-SEQUENCE tag, then the Media Sequence Number of the first Media Segment in the Media Playlist SHALL be considered to be 0. A client MUST NOT assume that segments with the same Media Sequence Number in different Media Playlists contain matching content (see Section 6.3.2).

A URI for a Media Segment is not required to contain its Media Sequence Number.

See Section 6.2.1 and Section 6.3.5 for more information on setting the EXT-X-MEDIA-SEQUENCE tag.

The EXT-X-MEDIA-SEQUENCE tag MUST appear before the first Media Segment in the Playlist.

4.4.3.3. EXT-X-DISCONTINUITY-SEQUENCE

The EXT-X-DISCONTINUITY-SEQUENCE tag allows synchronization between different Renditions of the same Variant Stream or different Variant Streams that have EXT-X-DISCONTINUITY tags in their Media Playlists.

Its format is:

#EXT-X-DISCONTINUITY-SEQUENCE:<number>

where number is a decimal-integer.

If the Media Playlist does not contain an EXT-X-DISCONTINUITY-SEQUENCE tag, then the Discontinuity Sequence Number of the first Media Segment in the Playlist SHALL be considered to be 0.

The EXT-X-DISCONTINUITY-SEQUENCE tag MUST appear before the first Media Segment in the Playlist.

The EXT-X-DISCONTINUITY-SEQUENCE tag MUST appear before any EXT-X-DISCONTINUITY tag.

See Section 6.2.1 and Section 6.2.2 for more information about setting the value of the EXT-X-DISCONTINUITY-SEQUENCE tag.

4.4.3.4. EXT-X-ENDLIST

The EXT-X-ENDLIST tag indicates that no more Media Segments will be added to the Media Playlist file. It MAY occur anywhere in the Media Playlist file. Its format is:

#EXT-X-ENDLIST

4.4.3.5. EXT-X-PLAYLIST-TYPE

The EXT-X-PLAYLIST-TYPE tag provides mutability information about the Media Playlist file. It applies to the entire Media Playlist file. It is OPTIONAL. Its format is:

#EXT-X-PLAYLIST-TYPE:<type-enum>

where type-enum is either EVENT or VOD.

Section 6.2.1 defines the implications of the EXT-X-PLAYLIST-TYPE tag.

If the EXT-X-PLAYLIST-TYPE value is EVENT, Media Segments can only be added to the end of the Media Playlist. If the EXT-X-PLAYLIST-TYPE value is Video On Demand (VOD), the Media Playlist cannot change.

If the EXT-X-PLAYLIST-TYPE tag is omitted from a Media Playlist, the Playlist can be updated according to the rules in Section 6.2.1 with no additional restrictions. For example, a live Playlist (Section 6.2.2) MAY be updated to remove Media Segments in the order that they appeared.

4.4.3.6. EXT-X-I-FRAMES-ONLY

The EXT-X-I-FRAMES-ONLY tag indicates that each Media Segment in the Playlist describes a single I-frame. I-frames are encoded video frames whose decoding does not depend on any other frame. I-frame Playlists can be used for trick play, such as fast forward, rapid reverse, and scrubbing.

The EXT-X-I-FRAMES-ONLY tag applies to the entire Playlist. Its format is:

#EXT-X-I-FRAMES-ONLY

In a Playlist with the EXT-X-I-FRAMES-ONLY tag, the Media Segment duration (EXTINF tag value) is the time between the presentation time of the I-frame in the Media Segment and the presentation time of the next I-frame in the Playlist, or the end of the presentation if it is the last I-frame in the Playlist.

Media resources containing I-frame segments MUST begin with either a Media Initialization Section (Section 3) or be accompanied by an EXT-X-MAP tag indicating the Media Initialization Section so that clients can load and decode I-frame segments in any order. The byte range of an I-frame segment with an EXT-X-BYTERANGE tag applied to it (Section 4.4.4.2) MUST NOT include its Media Initialization Section;

clients can assume that the Media Initialization Section is defined by the EXT-X-MAP tag, or is located between the start of the resource and the offset of the first I-frame segment in that resource.

Use of the EXT-X-I-FRAMES-ONLY REQUIRES a compatibility version number of 4 or greater.

4.4.3.7. EXT-X-PART-INF

The EXT-X-PART-INF tag provides information about the Partial Segments in the Playlist. It is REQUIRED if a Playlist contains one or more EXT-X-PART tags. Its format is:

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

The following attributes are defined:

PART-TARGET

The value is a decimal-floating-point number of seconds indicating the Part Target Duration. This attribute is REQUIRED.

4.4.3.8. EXT-X-SERVER-CONTROL

The EXT-X-SERVER-CONTROL tag allows the Server to indicate support for Delivery Directives (Section 6.2.5). Its format is:.

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

The following attributes are defined:

CAN-SKIP-UNTIL

Indicates that the Server can produce Playlist Delta Updates in response to the _HLS_skip Delivery Directive. Its value is the Skip Boundary, a decimal-floating-point number of seconds. The Skip Boundary MUST be at least six times the Target Duration.

This attribute is OPTIONAL. It MAY appear in any Media Playlist.

CAN-SKIP-DATERANGES

A value of YES indicates that the Server can produce Playlist Delta Updates that skip older EXT-X-DATERANGE tags in addition to Media Segments.

This attribute is OPTIONAL. It REQUIRES the presence of the CAN-SKIP-UNTIL attribute.

HOLD-BACK

The value is a decimal-floating-point number of seconds that indicates the server-recommended minimum distance from the end of the Playlist at which clients should begin to play or to which they should seek, unless PART-HOLD-BACK applies. Its value MUST be at least three times the Target Duration.

This attribute is OPTIONAL. Its absence implies a value of three times the Target Duration. It MAY appear in any Media Playlist.

PART-HOLD-BACK

The value is a decimal-floating-point number of seconds that indicates the server-recommended minimum distance from the end of the Playlist at which clients should begin to play or to which they should seek when playing in Low-Latency Mode. Its value MUST be at least twice the Part Target Duration. Its value SHOULD be at least three times the Part Target Duration. If different Renditions have different Part Target Durations then PART-HOLD-BACK SHOULD be at least three times the maximum Part Target Duration.

PART-HOLD-BACK is REQUIRED if the Playlist contains the EXT-X-PART-INF tag.

CAN-BLOCK-RELOAD

The value is an enumerated-string whose value is YES if the server supports Blocking Playlist Reload (Section 6.2.5.2). This attribute is OPTIONAL; its absence implies no support.

4.4.4. Media Segment Tags

Each Media Segment is specified by a series of Media Segment tags followed by a URI. Some Media Segment tags apply to just the next segment; others apply to all subsequent segments until another instance of the same tag.

A Media Segment tag MUST NOT appear in a Master Playlist. Clients MUST fail to parse Playlists that contain both Media Segment tags and Master Playlist tags (Section 4.4.6).

4.4.4.1. EXTINF

The EXTINF tag specifies the duration of a Media Segment. It applies only to the next Media Segment. This tag is REQUIRED for each Media Segment. Its format is:

#EXTINF:<duration>,[<title>]

where duration is a decimal-floating-point or decimal-integer number (as described in Section 4.2) that specifies the duration of the Media Segment in seconds. Durations SHOULD be decimal-floating-point, with enough accuracy to avoid perceptible error when segment durations are accumulated. However, if the compatibility version number is less than 3, durations MUST be integers. Durations that are reported as integers SHOULD be rounded to the nearest integer. The remainder of the line following the comma is an optional human-readable informative title of the Media Segment expressed as UTF-8 text.

4.4.4.2. EXT-X-BYTERANGE

The EXT-X-BYTERANGE tag indicates that a Media Segment is a sub-range of the resource identified by its URI. It applies only to the next URI line that follows it in the Playlist. Its format is:

#EXT-X-BYTERANGE:<n>[@<o>]

where n is a decimal-integer indicating the length of the sub-range in bytes. If present, o is a decimal-integer indicating the start of the sub-range, as a byte offset from the beginning of the resource. If o is not present, the sub-range begins at the next byte following the sub-range of the previous Media Segment.

If o is not present, a previous Media Segment MUST appear in the Playlist file and MUST be a sub-range of the same media resource, or the Media Segment is undefined and the client MUST fail to parse the Playlist.

A Media Segment without an EXT-X-BYTERANGE tag consists of the entire resource identified by its URI.

Use of the EXT-X-BYTERANGE tag REQUIRES a compatibility version number of 4 or greater.

4.4.4.3. EXT-X-DISCONTINUITY

The EXT-X-DISCONTINUITY tag indicates a discontinuity between the Media Segment that follows it and the one that preceded it.

Its format is:

#EXT-X-DISCONTINUITY

The EXT-X-DISCONTINUITY tag MUST be present if there is a change in any of the following characteristics:

- file format
- number, type, and identifiers of tracks
- timestamp sequence

The EXT-X-DISCONTINUITY tag SHOULD be present if there is a change in any of the following characteristics:

- encoding parameters
- encoding sequence

See Section 3, Section 6.2.1, and Section 6.3.3 for more information about the EXT-X-DISCONTINUITY tag.

4.4.4.4. EXT-X-KEY

Media Segments MAY be encrypted. The EXT-X-KEY tag specifies how to decrypt them. It applies to every Media Segment and to every Media Initialization Section declared by an EXT-X-MAP tag that appears between it and the next EXT-X-KEY tag in the Playlist file with the same KEYFORMAT attribute or a METHOD of NONE (or the end of the Playlist file). Any Media Segment or Media Initialization Section that precedes the first EXT-X-KEY tag is unencrypted. Two or more EXT-X-KEY tags with different KEYFORMAT attributes MAY apply to the same Media Segment if they ultimately produce the same decryption key. The format is:

#EXT-X-KEY:<attribute-list>

The following attributes are defined:

METHOD

The value is an enumerated-string that specifies the encryption method. This attribute is REQUIRED.

The methods defined are: NONE, AES-128, and SAMPLE-AES.

An encryption method of NONE means that Media Segments are not encrypted. If the encryption method is NONE, other attributes MUST NOT be present.

An encryption method of AES-128 signals that Media Segments are completely encrypted using the Advanced Encryption Standard (AES) [AES_128] with a 128-bit key, Cipher Block Chaining (CBC), and Public-Key Cryptography Standards #7 (PKCS7) padding [RFC5652]. CBC is restarted on each segment boundary, using either the Initialization Vector (IV) attribute value or the Media Sequence Number as the IV; see Section 5.2.

An alternative to whole-segment encryption is Sample Encryption. With Sample Encryption, only media sample data - such as audio packets or video frames - is encrypted. The rest of the Media Segment is unencrypted. Sample Encryption allows parts of the Segment to be processed without (or before) decrypting the media itself.

An encryption method of SAMPLE-AES means that the Media Segments are Sample Encrypted using the Advanced Encryption Standard [AES_128]. How these media streams are encrypted and encapsulated in a segment depends on the media encoding and the media format of the segment. fMP4 Media Segments are encrypted using the 'cbcs' scheme of Common Encryption [COMMON_ENC]. Encryption of other Media Segment formats containing H.264 [H_264], AAC [ISO_14496], AC-3 [AC_3], and Enhanced AC-3 [AC_3] media streams is described in the HTTP Live Streaming (HLS) Sample Encryption specification [SampleEnc]. The IV attribute MAY be present; see Section 5.2.

URI

The value is a quoted-string containing a URI that specifies how to obtain the key. This attribute is REQUIRED unless the METHOD is NONE.

IV

The value is a hexadecimal-sequence that specifies a 128-bit unsigned integer Initialization Vector to be used with the key. Use of the IV attribute REQUIRES a compatibility version number of 2 or greater. See Section 5.2 for when the IV attribute is used.

KEYFORMAT

The value is a quoted-string that specifies how the key is represented in the resource identified by the URI; see Section 5 for more detail. This attribute is OPTIONAL; its absence

indicates an implicit value of "identity". Use of the KEYFORMAT attribute REQUIRES a compatibility version number of 5 or greater.

KEYFORMATVERSIONS

The value is a quoted-string containing one or more positive integers separated by the "/" character (for example, "1", "1/2", or "1/2/5"). If more than one version of a particular KEYFORMAT is defined, this attribute can be used to indicate which version(s) this instance complies with. This attribute is OPTIONAL; if it is not present, its value is considered to be "1". Use of the KEYFORMATVERSIONS attribute REQUIRES a compatibility version number of 5 or greater.

If the Media Playlist file does not contain an EXT-X-KEY tag, then Media Segments are not encrypted.

See Section 5 for the format of the Key file, and Section 5.2, Section 6.2.3, and Section 6.3.6 for additional information on Media Segment encryption.

4.4.4.5. EXT-X-MAP

The EXT-X-MAP tag specifies how to obtain the Media Initialization Section (Section 3) required to parse the applicable Media Segments. It applies to every Media Segment that appears after it in the Playlist until the next EXT-X-MAP tag or until the end of the Playlist.

Its format is:

#EXT-X-MAP:<attribute-list>

The following attributes are defined:

URI

The value is a quoted-string containing a URI that identifies a resource that contains the Media Initialization Section. This attribute is REQUIRED.

BYTERANGE

The value is a quoted-string specifying a byte range into the resource identified by the URI attribute. This range SHOULD contain only the Media Initialization Section. The format of the byte range is described in Section 4.4.4.2. This attribute is

OPTIONAL; if it is not present, the byte range is the entire resource indicated by the URI.

An EXT-X-MAP tag SHOULD be supplied for Media Segments in Playlists with the EXT-X-I-FRAMES-ONLY tag when the first Media Segment (i.e., I-frame) in the Playlist (or the first segment following an EXT-X-DISCONTINUITY tag) does not immediately follow the Media Initialization Section at the beginning of its resource.

Use of the EXT-X-MAP tag in a Media Playlist that contains the EXT-X-I-FRAMES-ONLY tag REQUIRES a compatibility version number of 5 or greater. Use of the EXT-X-MAP tag in a Media Playlist that DOES NOT contain the EXT-X-I-FRAMES-ONLY tag REQUIRES a compatibility version number of 6 or greater.

If the Media Initialization Section declared by an EXT-X-MAP tag is encrypted with a METHOD of AES-128, the IV attribute of the EXT-X-KEY tag that applies to the EXT-X-MAP is REQUIRED.

4.4.4.6. EXT-X-PROGRAM-DATE-TIME

The EXT-X-PROGRAM-DATE-TIME tag associates the first sample of a Media Segment with an absolute date and/or time. It applies only to the next Media Segment. Its format is:

#EXT-X-PROGRAM-DATE-TIME:<date-time-msec>

where date-time-msec is an ISO/IEC 8601:2004 [ISO_8601] date/time representation, such as YYYY-MM-DDThh:mm:ss.SSSZ. It SHOULD indicate a time zone and fractional parts of seconds, to at least millisecond accuracy.

For example:

#EXT-X-PROGRAM-DATE-TIME:2010-02-19T14:54:23.031+08:00

See Section 6.2.1 and Section 6.3.3 for more information on the EXT-X-PROGRAM-DATE-TIME tag.

4.4.4.7. EXT-X-GAP

The EXT-X-GAP tag indicates that the segment URI to which it applies does not contain media data and SHOULD NOT be loaded by clients. It applies only to the next Media Segment.

Its format is:

#EXT-X-GAP

See Section 6.2.1 and Section 6.3.3 for more information on the EXT-X-GAP tag.

4.4.4.8. EXT-X-BITRATE

The EXT-X-BITRATE tag identifies the approximate segment bit rate of the Media Segment(s) to which it applies. It applies to every Media Segment between it and the next EXT-X-BITRATE tag in the Playlist file (or the end of the Playlist file) that does not have an EXT-X-BYTERANGE tag applied to it. Its format is:

#EXT-X-BITRATE:<rate>

where rate is a decimal-integer of kilobits per second.

This tag is OPTIONAL. If it is present then its value MUST be no less than 90% of the segment bit rate of each Media Segment to which it is applied and no greater than 110% of the segment bit rate of each Media Segment to which it is applied.

4.4.4.9. EXT-X-PART

The EXT-X-PART tag identifies a Partial Segment. It is OPTIONAL. Its format is:

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

The following attributes are defined:

URI

The value is the URI for the Partial Segment. This attribute is REQUIRED.

DURATION

The value is the duration of the Partial Segment as a decimal-floating-point number of seconds. This attribute is REQUIRED.

INDEPENDENT

The value is an enumerated-string whose value is YES if the Partial Segment contains an independent frame. This attribute is OPTIONAL; however every Partial Segment containing an independent frame SHOULD carry it to increase the efficiency with which clients can join and switch Renditions.

BYTERANGE

Indicates that the Partial Segment is a subrange of the resource specified by the URI attribute. The value is a quoted-string whose contents have the same format as the EXT-X-BYTERANGE tag: "<n>[@<o>]".

GAP

The value is an enumerated-string whose value is YES if the Partial Segment is not available. It is REQUIRED for such Partial Segments.

All Media Segment Tags (Section 4.4.4) except for EXT-X-BYTERANGE and EXT-X-GAP that are applied to a Parent Segment MUST appear before the first EXT-X-PART tag of that Parent Segment.

The duration of a Partial Segment MUST be less than or equal to the Part Target Duration. The duration of each Partial Segment MUST be at least 85% of the Part Target Duration, with the exception of Partial Segments with the INDEPENDENT=YES attribute and the final Partial Segment of any Parent Segment.

4.4.5. Media Metadata Tags

Media Metadata tags provide information about the playlist that is not associated with specific Media Segments. There MAY be more than one Media Metadata tag of each type in any Media Playlist. The only exception to this rule is EXT-X-SKIP, which MUST NOT appear more than once.

4.4.5.1. EXT-X-DATERANGE

The EXT-X-DATERANGE tag associates a Date Range (i.e., a range of time defined by a starting and ending date) with a set of attribute/value pairs. Its format is:

#EXT-X-DATERANGE:<attribute-list>

where the defined attributes are:

ID

A quoted-string that uniquely identifies a Date Range in the Playlist. This attribute is REQUIRED.

CLASS

A client-defined quoted-string that specifies some set of attributes and their associated value semantics. All Date Ranges

with the same CLASS attribute value MUST adhere to these semantics. This attribute is OPTIONAL.

START-DATE

A quoted-string containing the [ISO_8601] date/time at which the Date Range begins. This attribute is REQUIRED.

END-DATE

A quoted-string containing the [ISO_8601] date/time at which the Date Range ends. It MUST be equal to or later than the value of the START-DATE attribute. This attribute is OPTIONAL.

DURATION

The duration of the Date Range expressed as a decimal-floating-point number of seconds. It MUST NOT be negative. A single instant in time (e.g., crossing a finish line) SHOULD be represented with a duration of 0. This attribute is OPTIONAL.

PLANNED-DURATION

The expected duration of the Date Range expressed as a decimal-floating-point number of seconds. It MUST NOT be negative. This attribute SHOULD be used to indicate the expected duration of a Date Range whose actual duration is not yet known. It is OPTIONAL.

X-<client-attribute>

The "X-" prefix defines a namespace reserved for client-defined attributes. The client-attribute MUST be a legal AttributeName. Clients SHOULD use a reverse-DNS syntax when defining their own attribute names to avoid collisions. The attribute value MUST be a quoted-string, a hexadecimal-sequence, or a decimal-floating-point. An example of a client-defined attribute is X-COM-EXAMPLE-AD-ID="XYZ123". These attributes are OPTIONAL.

SCTE35-CMD, SCTE35-OUT, SCTE35-IN

Used to carry SCTE-35 data; see Section 4.4.5.1.1 for more information. These attributes are OPTIONAL.

END-ON-NEXT

An enumerated-string whose value MUST be YES. This attribute indicates that the end of the range containing it is equal to the

START-DATE of its Following Range. The Following Range is the Date Range of the same CLASS that has the earliest START-DATE after the START-DATE of the range in question. This attribute is OPTIONAL.

An EXT-X-DATERANGE tag with an END-ON-NEXT=YES attribute MUST have a CLASS attribute. Other EXT-X-DATERANGE tags with the same CLASS attribute MUST NOT specify Date Ranges that overlap.

An EXT-X-DATERANGE tag with an END-ON-NEXT=YES attribute MUST NOT contain DURATION or END-DATE attributes.

A Date Range with neither a DURATION, an END-DATE, nor an END-ON-NEXT=YES attribute has an unknown duration, even if it has a PLANNED-DURATION.

If a Playlist contains an EXT-X-DATERANGE tag, it MUST also contain at least one EXT-X-PROGRAM-DATE-TIME tag.

If a Playlist contains two EXT-X-DATERANGE tags with the same ID attribute value, then any AttributeName that appears in both tags MUST have the same AttributeValue.

If a Date Range contains both a DURATION attribute and an END-DATE attribute, the value of the END-DATE attribute MUST be equal to the value of the START-DATE attribute plus the value of the DURATION attribute.

Clients SHOULD ignore EXT-X-DATERANGE tags with illegal syntax.

4.4.5.1.1. Mapping SCTE-35 into EXT-X-DATERANGE

Splice information carried in source media according to the SCTE-35 specification [SCTE35] MAY be represented in a Media Playlist using EXT-X-DATERANGE tags.

Each SCTE-35 splice_info_section() containing a splice_null(), splice_schedule(), bandwidth_reservation(), or private_cmd() SHOULD be represented by an EXT-X-DATERANGE tag with an SCTE35-CMD attribute whose value is the big-endian binary representation of the splice_info_section(), expressed as a hexadecimal-sequence.

An SCTE-35 splice out/in pair signaled by a pair of splice_insert() commands SHOULD be represented by one or more EXT-X-DATERANGE tags carrying the same ID attribute, which MUST be unique to that splice out/in pair. The "out" splice_info_section() (with out_of_network_indicator set to 1) MUST be placed in an SCTE35-OUT attribute, with the same formatting as SCTE35-CMD. The "in"

splice_info_section() (with out_of_network_indicator set to 0) MUST be placed in an SCTE35-IN attribute, with the same formatting as SCTE35-CMD.

An SCTE-35 splice out/in pair signaled by a pair of time_signal() commands, each carrying a single segmentation_descriptor(), SHOULD be represented by one or more EXT-X-DATERANGE tags carrying the same ID attribute, which MUST be unique to that splice out/in pair. The "out" splice_info_section() MUST be placed in an SCTE35-OUT attribute; the "in" splice_info_section() MUST be placed in an SCTE35-IN attribute.

Different types of segmentation, as indicated by the segmentation_type_id in the segmentation_descriptor(), SHOULD be represented by separate EXT-X-DATERANGE tags, even if two or more segmentation_descriptor()s arrive in the same splice_info_section(). In that case, each EXT-X-DATERANGE tag will have an SCTE35-OUT, SCTE35-IN, or SCTE35-CMD attribute whose value is the entire splice_info_section().

An SCTE-35 time_signal() command that does not signal a splice out or in point SHOULD be represented by an EXT-X-DATERANGE tag with an SCTE35-CMD attribute.

The START-DATE of an EXT-X-DATERANGE tag containing an SCTE35-OUT attribute MUST be the date and time that corresponds to the program time of that splice.

The START-DATE of an EXT-X-DATERANGE tag containing an SCTE35-CMD MUST be the date and time specified by the splice_time() in the command or the program time at which the command appeared in the source stream if the command does not specify a splice_time().

An EXT-X-DATERANGE tag containing an SCTE35-OUT attribute MAY contain a PLANNED-DURATION attribute. Its value MUST be the planned duration of the splice.

The DURATION of an EXT-X-DATERANGE tag containing an SCTE35-IN attribute MUST be the actual (not planned) program duration between the corresponding out-point and that in-point.

The END-DATE of an EXT-X-DATERANGE tag containing an SCTE35-IN attribute MUST be the actual (not planned) program date and time of that in-point.

If the actual end date and time is not known when an SCTE35-OUT attribute is added to the Playlist, the DURATION attribute and the END-TIME attribute MUST NOT be present; the actual end date of the

splice SHOULD be signaled by another EXT-X-DATERANGE tag once it has been established.

A canceled splice SHOULD NOT appear in the Playlist as an EXT-X-DATERANGE tag.

An EXT-X-DATERANGE tag announcing a splice SHOULD be added to a Playlist at the same time as the last pre-splice Media Segment, or earlier if possible.

The ID attribute of an EXT-X-DATERANGE tag MAY contain a splice_event_id and/or a segmentation_event_id, but it MUST be unique in the Playlist. If there is a possibility that an SCTE-35 id will be reused, the ID attribute value MUST include disambiguation, such as a date or sequence number.

4.4.5.2. EXT-X-SKIP

A server produces a Playlist Delta Update (Section 6.2.5.1), by replacing tags earlier than the Skip Boundary with an EXT-X-SKIP tag.

When replacing Media Segments, the EXT-X-SKIP tag replaces the segment URI lines and all Media Segment Tags tags that are applied to those segments. This tag MUST NOT appear more than once in a Playlist

Its format is:

#EXT-X-SKIP:<attribute-list>

The following attributes are defined:

SKIPPED-SEGMENTS

The value is the count of Media Segments were replaced by the EXT-X-SKIP tag. This attribute is REQUIRED.

RECENTLY-REMOVED-DATERANGES

The value is a quoted-string consisting of a tab (0x9) delimited list of EXT-X-DATERANGE IDs that have been removed from the Playlist recently. See Section 6.2.5.1 for more information. This attribute is REQUIRED if the Client requested an update that skips EXT-X-DATERANGE tags. The quoted-string MAY be empty.

4.4.5.3. EXT-X-PRELOAD-HINT

The EXT-X-PRELOAD-HINT tag allows a Client loading media from a live stream to reduce the time to obtain a resource from the Server by issuing its request before the resource is available to be delivered. The server will hold onto the request ("block") until it can respond.

Its format is:

#EXT-X-PRELOAD-HINT:<attribute-list>

The following attributes are defined:

TYPE

The value is an enumerated-string that specifies the type of the hinted resource. If the value is PART, the resource is a Partial Segment. If the value is MAP, the resource is a Media Initialization Section. This attribute is REQUIRED.

URI

The value is a URI identifying the hinted resource. It MUST match the URI that will be subsequently added to the Playlist as a non-hinted resource (for example, the URI of an EXT-X-PART tag). The URI MAY be relative to the URI of the Playlist or it MAY be absolute. The hostname MAY differ from the hostname of the Playlist URI. This attribute is REQUIRED.

BYTERANGE-START

The value is a decimal-integer specifying the byte offset of the first byte of the hinted resource, from the beginning of the resource identified by the URI attribute. This attribute is OPTIONAL. Its absence implies a value of 0.

BYTERANGE-LENGTH

The value is a decimal-integer specifying the length of the hinted resource. This attribute is OPTIONAL. Its absence indicates that the last byte of the hinted resource is the last byte of the resource identified by the URI attribute. In this case, you SHOULD use the recommended last-byte-pos [RFC8673] value of 2^^53-1 (9007199254740991) in the HTTP Range request.

Note that when a hinted Partial Segment eventually appears in the Playlist as an EXT-X-PART tag, it MAY have a different Discontinuity Sequence Number, Media Initialization Section, or encryption

configuration. In other words, the Partial Segment can be preceded by an EXTINF tag indicating the end of the previous Parent Segment and an EXT-X-DISCONTINUITY, EXT-X-MAP, or EXT-X-KEY tag.

A Playlist containing an EXT-X-ENDLIST tag MUST NOT contain an EXT-X-PRELOAD-HINT tag.

4.4.5.4. EXT-X-RENDITION-REPORT

The EXT-X-RENDITION-REPORT tag carries information about an associated Rendition that is as up-to-date as the Playlist that contains it. Its format is:

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

The following attributes are defined:

URI

The value is the URI for the Media Playlist of the specified Rendition. It MUST be relative to the URI of the Media Playlist containing the EXT-X-RENDITION-REPORT tag. This attribute is REQUIRED.

LAST-MSN

The value is a decimal-integer specifying the Media Sequence Number of the last Media Segment currently in the specified Rendition. If the Rendition contains Partial Segments then this value is the Media Sequence Number of the last Partial Segment. This attribute is REQUIRED.

LAST-PART

The value is a decimal-integer that indicates the Part Index of the last Partial Segment currently in the specified Rendition whose Media Sequence Number is equal to the LAST-MSN attribute value. This attribute is REQUIRED if the Rendition contains a Partial Segment.

A server MAY omit adding an attribute to an EXT-X-RENDITION-REPORT tag - even a mandatory attribute - if its value is the same as that of the Rendition Report of the Media Playlist to which the EXT-X-RENDITION-REPORT tag is being added. Doing so reduces the size of the Rendition Report.

4.4.6. Master Playlist Tags

Master Playlist tags define the Variant Streams, Renditions, and other global parameters of the presentation.

Master Playlist tags MUST NOT appear in a Media Playlist; clients MUST fail to parse any Playlist that contains both a Master Playlist tag and either a Media Playlist tag or a Media Segment tag.

4.4.6.1. EXT-X-MEDIA

The EXT-X-MEDIA tag is used to relate Media Playlists that contain alternative Renditions (Section 4.4.6.2.1) of the same content. For example, three EXT-X-MEDIA tags can be used to identify audio-only Media Playlists that contain English, French, and Spanish Renditions of the same presentation. Or, two EXT-X-MEDIA tags can be used to identify video-only Media Playlists that show two different camera angles.

Its format is:

#EXT-X-MEDIA:<attribute-list>

The following attributes are defined:

TYPE

The value is an enumerated-string; valid strings are AUDIO, VIDEO, SUBTITLES, and CLOSED-CAPTIONS. This attribute is REQUIRED.

Typically, closed-caption [CEA608] media is carried in the video stream. Therefore, an EXT-X-MEDIA tag with TYPE of CLOSED-CAPTIONS does not specify a Rendition; the closed-caption media is present in the Media Segments of every video Rendition.

URI

The value is a quoted-string containing a URI that identifies the Media Playlist file. This attribute is OPTIONAL; see Section 4.4.6.2.1. If the TYPE is CLOSED-CAPTIONS, the URI attribute MUST NOT be present.

GROUP-ID

The value is a quoted-string that specifies the group to which the Rendition belongs. See Section 4.4.6.1.1. This attribute is REQUIRED.

LANGUAGE

The value is a quoted-string containing one of the standard Tags for Identifying Languages [RFC5646], which identifies the primary language used in the Rendition. This attribute is OPTIONAL.

ASSOC-LANGUAGE

The value is a quoted-string containing a language tag [RFC5646] that identifies a language that is associated with the Rendition. An associated language is often used in a different role than the language specified by the LANGUAGE attribute (e.g., written versus spoken, or a fallback dialect). This attribute is OPTIONAL.

NAME

The value is a quoted-string containing a human-readable description of the Rendition. If the LANGUAGE attribute is present, then this description SHOULD be in that language. This attribute is REQUIRED.

STABLE-RENDITION-ID

The value is a quoted-string which is a stable identifier for the URI within the Master Playlist. All characters in the quoted-string MUST be from the following set: [a..z], [A..Z], [0..9], '+', '/', '=', '.', '-', and '_'. This attribute is OPTIONAL.

The STABLE-RENDITION-ID allows the URI of a Rendition to change between two distinct downloads of the Master Playlist. IDs are matched using a byte-for-byte comparison.

DEFAULT

The value is an enumerated-string; valid strings are YES and NO. If the value is YES, then the client SHOULD play this Rendition of the content in the absence of information from the user indicating a different choice. This attribute is OPTIONAL. Its absence indicates an implicit value of NO.

AUTOSELECT

The value is an enumerated-string; valid strings are YES and NO. This attribute is OPTIONAL. Its absence indicates an implicit value of NO. If the value is YES, then the client MAY choose to play this Rendition in the absence of explicit user preference because it matches the current playback environment, such as chosen system language.

If the AUTOSELECT attribute is present, its value MUST be YES if the value of the DEFAULT attribute is YES.

FORCED

The value is an enumerated-string; valid strings are YES and NO. This attribute is OPTIONAL. Its absence indicates an implicit value of NO. The FORCED attribute MUST NOT be present unless the TYPE is SUBTITLES.

A value of YES indicates that the Rendition contains content that is considered essential to play. When selecting a FORCED Rendition, a client SHOULD choose the one that best matches the current playback environment (e.g., language).

A value of NO indicates that the Rendition contains content that is intended to be played in response to explicit user request.

INSTREAM-ID

The value is a quoted-string that specifies a Rendition within the segments in the Media Playlist. This attribute is REQUIRED if the TYPE attribute is CLOSED-CAPTIONS, in which case it MUST have one of the values: "CC1", "CC2", "CC3", "CC4", or "SERVICEn" where n MUST be an integer between 1 and 63 (e.g., "SERVICE9" or "SERVICE42").

The values "CC1", "CC2", "CC3", and "CC4" identify a Line 21 Data Services channel [CEA608]. The "SERVICE" values identify a Digital Television Closed Captioning [CEA708] service block number.

For all other TYPE values, the INSTREAM-ID MUST NOT be specified.

CHARACTERISTICS

The value is a quoted-string containing one or more Media Characteristic Tags (MCTs) separated by comma (,) characters. A Media Characteristic Tag has the same format as the payload of a media characteristic tag atom [MCT] This attribute is OPTIONAL. Each MCT indicates an individual characteristic of the Rendition.

A SUBTITLES Rendition MAY include the following characteristics: "public.accessibility.transcribes-spoken-dialog", "public.accessibility.describes-music-and-sound", and "public.easy-to-read" (which indicates that the subtitles have been edited for ease of reading).

An AUDIO Rendition MAY include the following characteristic: "public.accessibility.describes-video".

The CHARACTERISTICS attribute MAY include private MCTs.

CHANNELS

The value is a quoted-string that specifies an ordered, slash-separated ("/") list of parameters.

If the TYPE attribute is AUDIO, then the first parameter is a count of audio channels expressed as a decimal-integer, indicating the maximum number of independent, simultaneous audio channels present in any Media Segment in the Rendition. For example, an AC-3 5.1 Rendition would have a CHANNELS="6" attribute.

If the TYPE attribute is AUDIO, then the second parameter identifies the encoding of object-based audio used by the Rendition. This parameter is a comma-separated list of Audio Object Coding Identifiers. It is optional. An Audio Object Coding Identifier is a string containing characters from the set [A..Z], [0..9], and '-'. They are codec-specific. A parameter value of consisting solely of the dash character (0x2D) indicates that the audio is not object-based.

No other CHANNELS parameters are currently defined.

All audio EXT-X-MEDIA tags SHOULD have a CHANNELS attribute. If a Master Playlist contains two Renditions with the same NAME encoded with the same codec but a different number of channels, then the CHANNELS attribute is REQUIRED; otherwise, it is OPTIONAL.

The LANGUAGE and ASSOC-LANGUAGE attributes can be used, for example, to link Norwegian Renditions that use different spoken and written languages:

```
#EXT-X-MEDIA:TYPE=SUBTITLES,GROUP-ID="subtitles",
NAME="Bokmael",AUTOSELECT=YES,LANGUAGE="nb",
ASSOC-LANGUAGE="no",URI="nb-subtitles.m3u8"
```

This allows automatic selection of the Bokmael subtitles in this Media Playlist when the user picks an audio variant in Norwegian.

4.4.6.1.1. Rendition Groups

A set of one or more EXT-X-MEDIA tags with the same GROUP-ID value and the same TYPE value defines a Group of Renditions. Each member

of the Group MUST be an alternative Rendition of the same content; otherwise, playback errors can occur.

All EXT-X-MEDIA tags in a Playlist MUST meet the following constraints:

- All EXT-X-MEDIA tags in the same Group MUST have different NAME attributes.
- A Group MUST NOT have more than one member with a DEFAULT attribute of YES.
- Each EXT-X-MEDIA tag with an AUTOSELECT=YES attribute SHOULD have a combination of LANGUAGE [RFC5646], ASSOC-LANGUAGE, FORCED, and CHARACTERISTICS attributes that is distinct from those of other AUTOSELECT=YES members of its Group.

A Playlist MAY contain multiple Groups of the same TYPE in order to provide multiple encodings of that media type. If it does so, each Group of the same TYPE MUST have the same set of members, and each corresponding member MUST have identical attributes with the exception of the URI and CHANNELS attributes.

Each member in a Group of Renditions MAY have a different sample format. For example, an English Rendition can be encoded with AC-3 5.1 while a Spanish Rendition is encoded with AAC stereo. However, any EXT-X-STREAM-INF tag (Section 4.4.6.2) or EXT-X-I-FRAME-STREAM-INF tag (Section 4.4.6.3) that references such a Group MUST have a CODECS attribute that lists every sample format present in any Rendition in the Group, or client playback failures can occur. In the example above, the CODECS attribute would include "ac-3,mp4a.40.2".

4.4.6.2. EXT-X-STREAM-INF

The EXT-X-STREAM-INF tag specifies a Variant Stream, which is a set of Renditions that can be combined to play the presentation. The attributes of the tag provide information about the Variant Stream.

The URI line that follows the EXT-X-STREAM-INF tag specifies a Media Playlist that carries a Rendition of the Variant Stream. The URI line is REQUIRED. Clients that do not support multiple video Renditions SHOULD play this Rendition.

Its format is:

```
#EXT-X-STREAM-INF:<attribute-list>
<URI>
```

The following attributes are defined:

BANDWIDTH

The value is a decimal-integer of bits per second. It represents the peak segment bit rate of the Variant Stream.

If all the Media Segments in a Variant Stream have already been created, the BANDWIDTH value MUST be the largest sum of peak segment bit rates that is produced by any playable combination of Renditions. (For a Variant Stream with a single Media Playlist, this is just the peak segment bit rate of that Media Playlist.) An inaccurate value can cause playback stalls or prevent clients from playing the variant.

If the Master Playlist is to be made available before all Media Segments in the presentation have been encoded, the BANDWIDTH value SHOULD be the BANDWIDTH value of a representative period of similar content, encoded using the same settings.

Every EXT-X-STREAM-INF tag MUST include the BANDWIDTH attribute.

AVERAGE-BANDWIDTH

The value is a decimal-integer of bits per second. It represents the average segment bit rate of the Variant Stream.

If all the Media Segments in a Variant Stream have already been created, the AVERAGE-BANDWIDTH value MUST be the largest sum of average segment bit rates that is produced by any playable combination of Renditions. (For a Variant Stream with a single Media Playlist, this is just the average segment bit rate of that Media Playlist.) An inaccurate value can cause playback stalls or prevent clients from playing the variant.

If the Master Playlist is to be made available before all Media Segments in the presentation have been encoded, the AVERAGE-BANDWIDTH value SHOULD be the AVERAGE-BANDWIDTH value of a representative period of similar content, encoded using the same settings.

The AVERAGE-BANDWIDTH attribute is OPTIONAL.

SCORE

The value is a positive decimal-floating-point number. It is an abstract, relative measure of the playback quality-of-experience of the Variant Stream.

The value can be based on any metric or combination of metrics that can be consistently applied to all Variant Streams. The value SHOULD consider all media in the Variant Stream, including video, audio and subtitles. A Variant Stream with a SCORE attribute MUST be considered by the Playlist author to be more desirable than any Variant Stream with a lower SCORE attribute in the same Master Playlist.

The SCORE attribute is OPTIONAL, but if any Variant Stream contains the SCORE attribute, then all Variant Streams in the Master Playlist SHOULD have a SCORE attribute. See Section 6.3.1 for more information.

CODECS

The value is a quoted-string containing a comma-separated list of formats, where each format specifies a media sample type that is present in one or more Renditions specified by the Variant Stream. Valid format identifiers are those in the ISO Base Media File Format Name Space defined by "The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types" [RFC6381].

For example, a stream containing AAC low complexity (AAC-LC) audio and H.264 Main Profile Level 3.0 video would have a CODECS value of "mp4a.40.2,avc1.4d401e".

Note that if a Variant Stream specifies one or more Renditions that include IMSC subtitles, the CODECS attribute MUST indicate this with a format identifier such as "stpp.ttml.im1t".

Every EXT-X-STREAM-INF tag SHOULD include a CODECS attribute.

RESOLUTION

The value is a decimal-resolution describing the optimal pixel resolution at which to display all the video in the Variant Stream.

The RESOLUTION attribute is OPTIONAL but is recommended if the Variant Stream includes video.

FRAME-RATE

The value is a decimal-floating-point describing the maximum frame rate for all the video in the Variant Stream, rounded to three decimal places.

The FRAME-RATE attribute is OPTIONAL but is recommended if the Variant Stream includes video. The FRAME-RATE attribute SHOULD be included if any video in a Variant Stream exceeds 30 frames per second.

HDCP-LEVEL

The value is an enumerated-string; valid strings are TYPE-0, TYPE-1, and NONE. This attribute is advisory. A value of TYPE-0 indicates that the Variant Stream could fail to play unless the output is protected by High-bandwidth Digital Content Protection (HDCP) Type 0 [HDCP] or equivalent. A value of TYPE-1 indicates that the Variant Stream could fail to play unless the output is protected by HDCP Type 1 or equivalent. A value of NONE indicates that the content does not require output copy protection.

Encrypted Variant Streams with different HDCP levels SHOULD use different media encryption keys.

The HDCP-LEVEL attribute is OPTIONAL. It SHOULD be present if any content in the Variant Stream will fail to play without HDCP. Clients without output copy protection SHOULD NOT load a Variant Stream with an HDCP-LEVEL attribute unless its value is NONE.

ALLOWED-CPC

The ALLOWED-CPC attribute allows a server to indicate that the playback of a Variant Stream containing encrypted Media Segments is to be restricted to devices that guarantee a certain level of content protection robustness. Its value is a quoted-string containing a comma-separated list of entries. Each entry consists of a KEYFORMAT attribute value followed by a colon character (:) followed by a sequence of Content Protection Configuration (CPC) Labels separated by slash (/) characters. Each CPC Label is a string containing characters from the set [A..Z], [0..9], and '-'.

For example: ALLOWED-CPC="com.example.drm1:SMART-TV/PC, com.example.drm2:HW"

A CPC Label identifies a class of playback device that implements the KEYFORMAT with a certain level of content protection robustness. Each KEYFORMAT can define its own set of CPC Labels. The "identity" KEYFORMAT does not define any labels. A KEYFORMAT that defines CPC Labels SHOULD also specify its robustness requirements in a secure manner in each key response.

A client MAY play the Variant Stream if it implements one of the listed KEYFORMAT schemes with content protection robustness that

matches one or more of the CPC Labels in the list. If it does not match any of the CPC Labels then it SHOULD NOT attempt to play the Variant Stream.

The ALLOWED-CPC attribute is OPTIONAL. If it is not present or does not contain a particular KEYFORMAT then all clients that support that KEYFORMAT MAY play the Variant Stream.

VIDEO-RANGE

The value is an enumerated-string; valid strings are SDR, HLG and PQ.

The value MUST be SDR if the video in the Variant Stream is encoded using one of the following reference opto-electronic transfer characteristic functions specified by the TransferCharacteristics code point: [CICP] 1, 6, 13, 14, 15. Note that different TransferCharacteristics code points can use the same transfer function.

The value MUST be HLG if the video in the Variant Stream is encoded using a reference opto-electronic transfer characteristic function specified by the TransferCharacteristics code point 18, or consists of such video mixed with video qualifying as SDR (see above).

The value MUST be PQ if the video in the Variant Stream is encoded using a reference opto-electronic transfer characteristic function specified by the TransferCharacteristics code point 16, or consists of such video mixed with video qualifying as SDR or HLG (see above).

This attribute is OPTIONAL. Its absence implies a value of SDR. Clients that do not recognize the attribute value SHOULD NOT select the Variant Stream.

STABLE-VARIANT-ID

The value is a quoted-string which is a stable identifier for the URI within the Master Playlist. All characters in the quoted-string MUST be from the following set: [a..z], [A..Z], [0..9], '+', '/', '=', '.', '-', and '_'. This attribute is OPTIONAL.

The STABLE-VARIANT-ID allows the URI of the Variant Stream to change between two distinct downloads of the Master Playlist. IDs are matched using a byte-for-byte comparison.

AUDIO

The value is a quoted-string. It MUST match the value of the GROUP-ID attribute of an EXT-X-MEDIA tag elsewhere in the Master Playlist whose TYPE attribute is AUDIO. It indicates the set of audio Renditions that SHOULD be used when playing the presentation. See Section 4.4.6.2.1.

The AUDIO attribute is OPTIONAL.

VIDEO

The value is a quoted-string. It MUST match the value of the GROUP-ID attribute of an EXT-X-MEDIA tag elsewhere in the Master Playlist whose TYPE attribute is VIDEO. It indicates the set of video Renditions that SHOULD be used when playing the presentation. See Section 4.4.6.2.1.

The VIDEO attribute is OPTIONAL.

SUBTITLES

The value is a quoted-string. It MUST match the value of the GROUP-ID attribute of an EXT-X-MEDIA tag elsewhere in the Master Playlist whose TYPE attribute is SUBTITLES. It indicates the set of subtitle Renditions that can be used when playing the presentation. See Section 4.4.6.2.1.

The SUBTITLES attribute is OPTIONAL.

CLOSED-CAPTIONS

The value can be either a quoted-string or an enumerated-string with the value NONE. If the value is a quoted-string, it MUST match the value of the GROUP-ID attribute of an EXT-X-MEDIA tag elsewhere in the Playlist whose TYPE attribute is CLOSED-CAPTIONS, and it indicates the set of closed-caption Renditions that can be used when playing the presentation. See Section 4.4.6.2.1.

If the value is the enumerated-string value NONE, all EXT-X-STREAM-INF tags MUST have this attribute with a value of NONE, indicating that there are no closed captions in any Variant Stream in the Master Playlist. Having closed captions in one Variant Stream but not another can trigger playback inconsistencies.

The CLOSED-CAPTIONS attribute is OPTIONAL.

4.4.6.2.1. Alternative Renditions

When an EXT-X-STREAM-INF tag contains an AUDIO, VIDEO, SUBTITLES, or CLOSED-CAPTIONS attribute, it indicates that alternative Renditions of the content are available for playback of that Variant Stream.

When defining alternative Renditions, the following constraints MUST be met to prevent client playback errors:

- All playable combinations of Renditions associated with an EXT-X-STREAM-INF tag MUST have an aggregate bandwidth less than or equal to the BANDWIDTH attribute of the EXT-X-STREAM-INF tag.

- If an EXT-X-STREAM-INF tag contains a RESOLUTION attribute and a VIDEO attribute, then every alternative video Rendition MUST have an optimal display resolution matching the value of the RESOLUTION attribute.

- Every alternative Rendition associated with an EXT-X-STREAM-INF tag MUST meet the constraints for a Variant Stream described in Section 6.2.4.

The URI attribute of the EXT-X-MEDIA tag is REQUIRED if the media type is SUBTITLES, but OPTIONAL if the media type is VIDEO or AUDIO. If the media type is VIDEO or AUDIO, a missing URI attribute indicates that the media data for this Rendition is included in the Media Playlist of any EXT-X-STREAM-INF tag referencing this EXT-X-MEDIA tag. If the media TYPE is AUDIO and the URI attribute is missing, clients MUST assume that the audio data for this Rendition is present in every video Rendition specified by the EXT-X-STREAM-INF tag.

The URI attribute of the EXT-X-MEDIA tag MUST NOT be included if the media type is CLOSED-CAPTIONS.

4.4.6.3. EXT-X-I-FRAME-STREAM-INF

The EXT-X-I-FRAME-STREAM-INF tag identifies a Media Playlist file containing the I-frames of a multimedia presentation. It stands alone, in that it does not apply to a particular URI in the Master Playlist. Its format is:

#EXT-X-I-FRAME-STREAM-INF:<attribute-list>

All attributes defined for the EXT-X-STREAM-INF tag (Section 4.4.6.2) are also defined for the EXT-X-I-FRAME-STREAM-INF tag, except for the FRAME-RATE, AUDIO, SUBTITLES, and CLOSED-CAPTIONS attributes. In addition, the following attribute is defined:

URI

The value is a quoted-string containing a URI that identifies the I-frame Media Playlist file. That Playlist file MUST contain an EXT-X-I-FRAMES-ONLY tag.

Every EXT-X-I-FRAME-STREAM-INF tag MUST include a BANDWIDTH attribute and a URI attribute.

The provisions in Section 4.4.6.2.1 also apply to EXT-X-I-FRAME-STREAM-INF tags with a VIDEO attribute.

A Master Playlist that specifies alternative VIDEO Renditions and I-frame Playlists SHOULD include an alternative I-frame VIDEO Rendition for each regular VIDEO Rendition, with the same NAME and LANGUAGE attributes.

4.4.6.4. EXT-X-SESSION-DATA

The EXT-X-SESSION-DATA tag allows arbitrary session data to be carried in a Master Playlist.

Its format is:

#EXT-X-SESSION-DATA:<attribute-list>

The following attributes are defined:

DATA-ID

The value of DATA-ID is a quoted-string that identifies a particular data value. The DATA-ID SHOULD conform to a reverse DNS naming convention, such as "com.example.movie.title"; however, there is no central registration authority, so Playlist authors SHOULD take care to choose a value that is unlikely to collide with others. This attribute is REQUIRED.

VALUE

VALUE is a quoted-string. It contains the data identified by DATA-ID. If the LANGUAGE is specified, VALUE SHOULD contain a human-readable string written in the specified language.

URI

The value is a quoted-string containing a URI. The resource identified by the URI MUST be formatted as JSON [RFC8259]; otherwise, clients may fail to interpret the resource.

LANGUAGE

The value is a quoted-string containing a language tag [RFC5646] that identifies the language of the VALUE. This attribute is OPTIONAL.

Each EXT-X-SESSION-DATA tag MUST contain either a VALUE or URI attribute, but not both.

A Playlist MAY contain multiple EXT-X-SESSION-DATA tags with the same DATA-ID attribute. A Playlist MUST NOT contain more than one EXT-X-SESSION-DATA tag with the same DATA-ID attribute and the same LANGUAGE attribute.

4.4.6.5. EXT-X-SESSION-KEY

The EXT-X-SESSION-KEY tag allows encryption keys from Media Playlists to be specified in a Master Playlist. This allows the client to preload these keys without having to read the Media Playlist(s) first.

Its format is:

#EXT-X-SESSION-KEY:<attribute-list>

All attributes defined for the EXT-X-KEY tag (Section 4.4.4.4) are also defined for the EXT-X-SESSION-KEY, except that the value of the METHOD attribute MUST NOT be NONE. If an EXT-X-SESSION-KEY is used, the values of the METHOD, KEYFORMAT, and KEYFORMATVERSIONS attributes MUST match any EXT-X-KEY with the same URI value.

EXT-X-SESSION-KEY tags SHOULD be added if multiple Variant Streams or Renditions use the same encryption keys and formats. An EXT-X-SESSION-KEY tag is not associated with any particular Media Playlist.

A Master Playlist MUST NOT contain more than one EXT-X-SESSION-KEY tag with the same METHOD, URI, IV, KEYFORMAT, and KEYFORMATVERSIONS attribute values.

The EXT-X-SESSION-KEY tag is optional.

5. Key Files

5.1. Structure of Key Files

An EXT-X-KEY tag with a URI attribute identifies a Key file. A Key file contains a cipher key that can decrypt Media Segments in the Playlist.

[AES_128] encryption uses 16-octet keys. If the KEYFORMAT of an EXT-X-KEY tag is "identity", the Key file is a single packed array of 16 octets in binary format.

5.2. IV for AES-128

[AES_128] REQUIRES the same 16-octet IV to be supplied when encrypting and decrypting. Varying this IV increases the strength of the cipher.

An IV attribute on an EXT-X-KEY tag with a KEYFORMAT of "identity" specifies an IV that can be used when decrypting Media Segments encrypted with that Key file. IV values for AES-128 are 128-bit numbers.

An EXT-X-KEY tag with a KEYFORMAT of "identity" that does not have an IV attribute indicates that the Media Sequence Number is to be used as the IV when decrypting a Media Segment, by putting its big-endian binary representation into a 16-octet (128-bit) buffer and padding (on the left) with zeros.

6. Client/Server Responsibilities

6.1. Introduction

This section describes how the server generates the Playlist and Media Segments and how the client should download them for playback.

6.2. Server Responsibilities

6.2.1. General Server Responsibilities

The production of the source media is outside the scope of this document, which simply presumes a source of continuous encoded media containing the presentation.

The server MUST divide the source media into individual Media Segments whose duration (when rounded to a whole second) is less than or equal to the Target Duration. Segments longer than that can trigger playback stalls and other errors.

The server SHOULD attempt to divide the source media at points that support effective decode of individual Media Segments, such as on packet and key frame boundaries.

The server MUST create a URI for every Media Segment that enables its clients to obtain the segment data. If a server supports partial loading of resources (e.g., via HTTP Range requests), it MAY specify

segments as sub-ranges of larger resources using the EXT-X-BYTERANGE tag.

The absence of media data (due to, for example, the temporary unavailability of an encoder) SHOULD be signaled by adding one or more Media Segments to the Playlist whose Segment durations add up to the duration of absent media; these Media Segments MUST have EXT-X-GAP tags applied to them. Similarly, such Partial Segments MUST have a GAP=YES attribute. Attempting to download these segments MAY produce an error, such as HTTP 404 or 410.

A Media Segment MUST be available for immediate download at the full speed of the link to the Client when it is added to a Playlist unless it has been marked with an EXT-X-GAP tag; otherwise playback errors can occur. Once download starts, its transfer rate SHOULD NOT be constrained by the segment production process.

A Partial Segment MUST be similarly available at the time it is added to a Playlist.

HTTP servers SHOULD transfer text files -- such as Playlists and WebVTT segments -- using the "gzip" Content-Encoding if the client indicates that it is prepared to accept it.

The server must create a Media Playlist file (Section 4) that contains a URI for each Media Segment that the server wishes to make available, in the order in which they are to be played.

The value of the EXT-X-VERSION tag (Section 4.4.1.2) SHOULD NOT be greater than what is required for the tags and attributes in the Playlist (see Section 7).

Changes to the Playlist file MUST be made atomically from the point of view of the clients, or playback errors MAY occur.

The server MUST NOT change the Media Playlist file, except to:

- Append lines to it (Section 6.2.1).
- Remove Media Segment URIs from the Playlist in the order that they appear, along with any tags that apply only to those segments (Section 6.2.2).
- Remove Media Metadata tags that no longer apply to the presentation (Section 6.2.1).
- Remove EXT-X-PART tags no longer at the live edge (Section 6.2.2).

Increment the value of the EXT-X-MEDIA-SEQUENCE or EXT-X-DISCONTINUITY-SEQUENCE tags (Section 6.2.2).

Add an EXT-X-ENDLIST tag to the Playlist (Section 6.2.1).

A Media Playlist has further constraints on its updates if it contains an EXT-X-PLAYLIST-TYPE tag. An EXT-X-PLAYLIST-TYPE tag with a value of VOD indicates that the Playlist file MUST NOT change. An EXT-X-PLAYLIST-TYPE tag with a value of EVENT indicates that the Server MUST NOT change or remove any part of the Playlist file, with the exception of EXT-X-PART tags and Media Metadata tags as described above; the Server MAY append lines to the Playlist.

The value of the EXT-X-TARGETDURATION tag in the Media Playlist MUST NOT change. A typical Target Duration is 6 seconds.

Playlist changes other than those allowed here can trigger playback errors and inconsistent client behavior.

Each Media Segment in a Media Playlist has an integer Discontinuity Sequence Number. The Discontinuity Sequence Number can be used in addition to the timestamps within the media to synchronize Media Segments across different Renditions.

A segment’s Discontinuity Sequence Number is the value of the EXT-X-DISCONTINUITY-SEQUENCE tag (or zero if none) plus the number of EXT-X-DISCONTINUITY tags in the Playlist preceding the URI line of the segment.

The server MAY associate an absolute date and time with a Media Segment by applying an EXT-X-PROGRAM-DATE-TIME tag to it. This defines an informative mapping of the (wall-clock) date and time specified by the tag to the first media timestamp in the segment, which may be used as a basis for seeking, for display, or for other purposes. If a server provides this mapping, it SHOULD apply an EXT-X-PROGRAM-DATE-TIME tag to every segment that has an EXT-X-DISCONTINUITY tag applied to it.

The Server MUST NOT add any EXT-X-PROGRAM-DATE-TIME tag to a Playlist that would cause the mapping between program date and Media Segment to become ambiguous.

When applied to live content, a reasonable default for the EXT-X-PROGRAM-DATE-TIME tag is the date and time that the content was captured (recorded).

The server MUST NOT remove an EXT-X-DATERANGE tag from a Playlist if any date in the range maps to a Media Segment in the Playlist.

The server MUST NOT reuse the ID attribute value of an EXT-X-DATERANGE tag for any new Date Range in the same Playlist.

Once the Following Range of a Date Range with an END-ON-NEXT=YES attribute is added to a Playlist, the Server MUST NOT subsequently add a Date Range with the same CLASS attribute whose START-DATE is between that of the END-ON-NEXT=YES range and its Following Range.

For Date Ranges with a PLANNED-DURATION attribute, the Server SHOULD signal the actual end of the range once it has been established. It can do so by adding another EXT-X-DATERANGE tag with the same ID attribute value and either a DURATION or an END-DATE attribute or, if the Date Range has an END-ON-NEXT=YES attribute, by adding a Following Range.

If the Media Playlist contains the final Media Segment of the presentation, then the Playlist file MUST contain the EXT-X-ENDLIST tag; this allows clients to minimize unproductive Playlist reloads.

If a Media Playlist does not contain the EXT-X-ENDLIST tag, the server MUST make a new version of the Playlist file available that contains at least one new Media Segment. It MUST be made available no later than 1.5 times the Target Duration after the previous time the Playlist was updated with a Media Segment. This allows clients to utilize the network efficiently.

If a Media Playlist without an EXT-X-ENDLIST tag contains Partial Segments, the Server MUST add a new Partial Segment to the Playlist within one Part Target Duration after it added the previous Partial Segment.

If the server wishes to remove an entire presentation, it SHOULD provide a clear indication to clients that the Playlist file is no longer available (e.g., with an HTTP 404 or 410 response). It MUST ensure that all Media Segments in the Playlist file remain available to clients for at least the duration of the Playlist file at the time of removal to prevent interruption of in-progress playback.

6.2.2. Live Playlists

The server MAY limit the availability of Media Segments by removing Media Segments from the Playlist file (Section 6.2.1). If Media Segments are to be removed, the Playlist file MUST contain an EXT-X-MEDIA-SEQUENCE tag. Its value MUST be incremented by 1 for every Media Segment that is removed from the Playlist file; it MUST NOT decrease or wrap. Clients can malfunction if each Media Segment does not have a consistent, unique Media Sequence Number.

EXT-X-PART tags SHOULD be removed from the Playlist after they are greater than three Target Durations from the end of the Playlist.

Media Segments and EXT-X-PART tags MUST be removed from the Playlist in the order that they appear in the Playlist; otherwise, client playback can malfunction.

The server MUST NOT remove a Media Segment from a Playlist file without an EXT-X-ENDLIST tag if that would produce a Playlist whose duration is less than three times the Target Duration. Doing so can trigger playback stalls.

The Availability Duration of a Media Segment is the duration of the segment plus the duration of the longest-duration Playlist distributed by the server containing that segment. If the server removes a Media Segment URI from a Playlist that contains an EXT-X-ENDLIST tag, clients MUST be able to download the corresponding Media Segment until the time of removal plus the segment's Availability Duration. If the server removes a Media Segment URI from a Playlist that does not contain an EXT-X-ENDLIST tag, clients MUST be able to download the segment until the time at which it first appeared in the Playlist plus the segment's Availability Duration.

If the server wishes to remove segments from a Media Playlist containing an EXT-X-DISCONTINUITY tag, the Media Playlist MUST contain an EXT-X-DISCONTINUITY-SEQUENCE tag. Without the EXT-X-DISCONTINUITY-SEQUENCE tag, it can be impossible for a client to locate corresponding segments between Renditions.

If the server removes an EXT-X-DISCONTINUITY tag from the Media Playlist, it MUST increment the value of the EXT-X-DISCONTINUITY-SEQUENCE tag so that the Discontinuity Sequence Numbers of the segments still in the Media Playlist remain unchanged. The value of the EXT-X-DISCONTINUITY-SEQUENCE tag MUST NOT decrease or wrap. Clients can malfunction if each Media Segment does not have a consistent Discontinuity Sequence Number.

If a server plans to remove a Media Segment after it is delivered to clients over HTTP, it SHOULD ensure that the HTTP response contains an Expires header that reflects the planned time-to-live.

A Live Playlist MUST NOT contain the EXT-X-PLAYLIST-TYPE tag, as no value of that tag allows Media Segments to be removed.

6.2.3. Encrypting Media Segments

Media Segments MAY be encrypted. Every encrypted Media Segment MUST have an EXT-X-KEY tag (Section 4.4.4.4) applied to it with a URI that the client can use to obtain a Key file (Section 5) containing the decryption key.

A Media Segment can only be encrypted with one encryption METHOD, using one encryption key and IV. However, a server MAY offer multiple ways to retrieve that key by providing multiple EXT-X-KEY tags, each with a different KEYFORMAT attribute value.

The server MAY set the HTTP Expires header in the key response to indicate the duration for which the key can be cached.

Any unencrypted Media Segment in a Playlist MUST be in the scope of an EXT-X-KEY tag that specifies an encryption METHOD of NONE or precedes the first EXT-X-KEY tag. Otherwise, the client will misinterpret those segments as encrypted.

If the encryption METHOD is AES-128 and the Playlist does not contain the EXT-X-I-FRAMES-ONLY tag, AES encryption as described in Section 4.4.4.4 SHALL be applied to individual Media Segments.

If the encryption METHOD is AES-128 and the Playlist contains an EXT-X-I-FRAMES-ONLY tag, the entire resource MUST be encrypted using AES-128 CBC with PKCS7 padding [RFC5652]. Encryption MAY be restarted on 16-byte block boundaries, unless the first block contains an I-frame. The IV used for encryption MUST be either the Media Sequence Number of the Media Segment or the value of the IV attribute of the EXT-X-KEY tag, as described in Section 5.2. These constraints allow a client to load and decrypt individual I-frames specified as sub-ranges of regular encrypted Media Segments, and their Media Initialization Sections.

If the encryption METHOD indicates Sample Encryption, media samples MAY be encrypted prior to encapsulation in a Media Segment.

The server MUST NOT remove an EXT-X-KEY tag from the Playlist file if it applies to any Media Segment in the Playlist file, or clients who subsequently load that Playlist will be unable to decrypt those Media Segments.

6.2.4. Providing Variant Streams

A server MAY offer multiple Media Playlist files to provide different encodings of the same presentation. If it does so, it SHOULD provide

a Master Playlist file that lists each Variant Stream to allow clients to switch between encodings dynamically.

Master Playlists describe regular Variant Streams with EXT-X-STREAM-INF tags and I-frame Variant Streams with EXT-X-I-FRAME-STREAM-INF tags.

If an EXT-X-STREAM-INF tag or EXT-X-I-FRAME-STREAM-INF tag contains the CODECS attribute, the attribute value MUST include every media format [RFC6381] present in any Media Segment in any of the Renditions specified by the Variant Stream.

The server MUST meet the following constraints when producing Variant Streams in order to allow clients to switch between them seamlessly:

Each Variant Stream MUST present the same content.

Matching content in Variant Streams MUST have matching timestamps. This allows clients to synchronize the media.

Matching content in Variant Streams MUST have matching Discontinuity Sequence Numbers (see Section 4.4.3.3).

Each Media Playlist in each Variant Stream MUST have the same Target Duration. The only exceptions are SUBTITLES Renditions and Media Playlists containing an EXT-X-I-FRAMES-ONLY tag, which MAY have different Target Durations if they have an EXT-X-PLAYLIST-TYPE of VOD.

Content that appears in a Media Playlist of one Variant Stream but not in another MUST appear either at the beginning or at the end of the Media Playlist file and MUST NOT be longer than the Target Duration.

If any Media Playlists have an EXT-X-PLAYLIST-TYPE tag, all Media Playlists MUST have an EXT-X-PLAYLIST-TYPE tag with the same value.

If the Playlist contains an EXT-X-PLAYLIST-TYPE tag with the value of VOD, the first segment of every Media Playlist in every Variant Stream MUST start at the same media timestamp.

If any Media Playlist in a Master Playlist contains an EXT-X-PROGRAM-DATE-TIME tag, then all Media Playlists in that Master Playlist MUST contain EXT-X-PROGRAM-DATE-TIME tags with consistent mappings of date and time to media timestamps.

Each Variant Stream MUST contain the same set of Date Ranges. The EXT-X-DATERANGE tags of corresponding Date Ranges MUST have the same ID attribute value and contain the same set of attribute/value pairs.

If any Media Playlist in a Master Playlist contains an EXT-X-SERVER-CONTROL tag, then all Media Playlists in that Master Playlist MUST contain that tag, with the same attributes and values.

In addition, for broadest compatibility, Variant Streams SHOULD contain the same encoded audio bitstream. This allows clients to switch between Variant Streams without audible glitching.

The rules for Variant Streams also apply to alternative Renditions (see Section 4.4.6.2.1).

6.2.5. Delivery Directives Interface

A server MAY provide a set of services to its clients by implementing support for Delivery Directives. Delivery Directives are transmitted by the Client to the Server as Query Parameters in Playlist request URIs.

Servers advertise the availability of Delivery Directives using the EXT-X-SERVER-CONTROL tag (Section 4.4.3.8).

Currently-defined Delivery Directives are _HLS_skip, _HLS_msn and _HLS_part.

6.2.5.1. Playlist Delta Updates

Live presentations involve frequent Playlist downloads. When Playlists are large and a Client already has the previous version, the transfer cost can be reduced considerably by sending only the newest information in response to a Playlist update request.

A Server advertises support for Playlist Delta Updates that skip older Media Segments by adding the CAN-SKIP-UNTIL attribute to the EXT-X-SERVER-CONTROL tag. A Server can also offer support for Playlist Delta Updates that skip older EXT-X-DATERANGE tags by adding the CAN-SKIP-DATERANGES attribute to the EXT-X-SERVER-CONTROL tag.

When a Server receives a request for a Playlist containing the CAN-SKIP-UNTIL attribute but no EXT-X-ENDLIST tag, and the requested URI contains an _HLS_skip directive whose value is YES or v2, it MUST respond with a Playlist Delta Update.

The Playlist Delta Update is a version of the Playlist in which Media Segments that are further from the end of the Playlist than the Skip Boundary (Section 4.4.3.8), as well as their associated tags, are replaced by an EXT-X-SKIP tag (Section 4.4.5.2).

When the _HLS_skip directive has a value of v2, the Playlist Delta Update additionally MUST NOT contain EXT-X-DATERANGE tags that were added to the Playlist more than CAN-SKIP-UNTIL seconds before the Playlist request. The RECENTLY-REMOVED-DATERANGES attribute of the EXT-X-SKIP tag MUST list the date ranges that were removed from the Playlist within CAN-SKIP-UNTIL seconds of the Playlist request.

All tags that were not skipped MUST remain in the Playlist Delta Update.

A Server MUST ignore the _HLS_skip directive if the Playlist does not contain the CAN-SKIP-UNTIL attribute, or if it contains an EXT-X-ENDLIST tag.

6.2.5.2. Blocking Playlist Reload

A Server MAY offer Blocking Playlist Reloads, which enable immediate client discovery of Playlist updates as an alternative to polling.

A Server advertises support for Blocking Playlist Reload by adding the CAN-BLOCK-RELOAD=YES attribute to the EXT-X-SERVER-CONTROL tag.

A Client requests a Blocking Playlist Reload using an _HLS_msn directive with a decimal-integer value M. When the Playlist URI contains an _HLS_msn directive and no _HLS_part directive, the Server MUST defer responding to the request until the Playlist contains a Media Segment with a Media Sequence Number of M or later or it responds with an error.

The Playlist URI MAY also contain an _HLS_part directive with a decimal-integer value N. When the Playlist URI contains both an _HLS_msn directive and an _HLS_part directive, the Server MUST defer responding to the request until the Playlist contains the Partial Segment with Part Index N and with a Media Sequence Number of M or later or it responds with an error.

If the Client requests a Part Index greater than that of the final Partial Segment of the Parent Segment, the Server MUST treat the request as one for Part Index 0 of the following Parent Segment.

The Server MUST deliver the entire Playlist, even if the requested Media Segment is not the last one in the Playlist, and even if it is no longer in the Playlist.

A Server MUST ignore _HLS_msn and _HLS_part if the Playlist contains an EXT-X-ENDLIST tag.

If the _HLS_msn is greater than the Media Sequence Number of the last Media Segment in the current Playlist plus two, or if the _HLS_part exceeds the last Partial Segment in the current Playlist by the Advance Part Limit, then the server SHOULD immediately return Bad Request, such as HTTP 400. The Advance Part Limit is three divided by the Part Target Duration if the Part Target Duration is less than one second, or three otherwise.

If the Playlist URI contains an _HLS_part directive but no _HLS_msn directive, the Server MUST return Bad Request, such as HTTP 400.

A Server that cannot provide the requested Playlist after blocking for more than three Target Durations SHOULD return Service Unavailable, such as HTTP 503.

6.2.6. Providing Preload Hints

The Server MAY add EXT-X-PRELOAD-HINT tags (Section 4.4.5.3) to the Playlist to allow Clients playing the stream to request upcoming resources in advance.

A hinted resource MUST be available for request when its EXT-X-PRELOAD-HINT tag is added to the Playlist.

When processing requests for a URI or a byte range of a URI that includes one or more Partial Segments that are not yet completely available to be sent - such as requests made in response to an EXT-X-PRELOAD-HINT tag - the server MUST refrain from transmitting any bytes belonging to a Partial Segment until all bytes of that Partial Segment can be transmitted at the full speed of the link to the client. If the requested range includes more than one Partial Segment then the server MUST enforce this delivery guarantee for each Partial Segment in turn. This enables the client to perform accurate Adaptive Bit Rate (ABR) measurements.

The Server SHOULD NOT hint a byte range that it does not expect its clients to require in the near term.

The server SHOULD respond with "Not Found" (such as HTTP 404) to a request for a resource that it cannot find and that is not specified by an EXT-X-PRELOAD-HINT tag in an active Media Playlist.

A server MAY choose not to publish previously-hinted resources if the planned segmentation changes, such as the case of early return from

an ad. The server SHOULD respond to client requests for those resources with "Not Found" (such as HTTP 404).

If a Partial Segment is created as a subrange of a larger resource and its length is not known at the time that its hint is added to the Playlist, the BYTERANGE-LENGTH attribute SHOULD be omitted. The BYTERANGE-OFFSET SHOULD indicate the Partial Segment's starting offset into the larger resource.

The Server SHOULD NOT add more than one EXT-X-PRELOAD-HINT tag with the same TYPE to a Playlist.

6.3. Client Responsibilities

6.3.1. General Client Responsibilities

How the client obtains the URI to the Playlist file is outside the scope of this document; it is presumed to have done so.

The client obtains the Playlist file from the URI. If the Playlist file so obtained is a Master Playlist, the client can select a Variant Stream to load from the Master Playlist.

Clients MUST ensure that loaded Playlists comply with Section 4 and that the EXT-X-VERSION tag, if present, specifies a protocol version supported by the client; if either check fails, the client MUST NOT attempt to use the Playlist, or unintended behavior could occur.

When parsing Playlist elements that are subject to variable substitution, a Variable Reference whose Variable Name has been provided by an EXT-X-DEFINE tag that precedes the Variable Reference MUST be replaced by the corresponding Variable Value. Such replacements themselves are NOT subject to variable substitution.

When parsing Playlist elements that are subject to variable substitution, a Variable Reference whose Variable Name has NOT been provided by an EXT-X-DEFINE tag preceding the Variable Reference MUST trigger a parsing error.

If any URI element in a Playlist contains an URI scheme that the client cannot handle, the client MUST stop playback. All clients MUST support HTTP schemes.

To support forward compatibility, when parsing Playlists, clients MUST:

- ignore any unrecognized tags.

- ignore any attribute/value pair with an unrecognized AttributeName.

- ignore any tag containing an attribute/value pair of type enumerated-string whose AttributeName is recognized but whose AttributeValue is not recognized, unless the definition of the attribute says otherwise.

When identifying playable Renditions, Clients SHOULD consider an audio Rendition having unrecognized CHANNELS parameters to be playable if its associated CODECS attribute is supported. However, an equivalent Rendition with the same audio codec and recognized CHANNELS parameters SHOULD be preferred if it is present in the Master Playlist.

When all Variant Streams have a SCORE attribute, the client SHOULD use the SCORE value to choose a Variant Stream after all other playability constraints have been applied. If several Variant Streams have the highest SCORE value then other criteria MAY be used to chose among them.

Algorithms used by the client to switch between Variant Streams are beyond the scope of this document.

6.3.2. Loading the Media Playlist File

Every time a Media Playlist is loaded or reloaded from a Playlist URI, the client MUST determine the next Media Segment to load, as described in Section 6.3.5, if it intends to play the presentation normally (i.e., in Playlist order at the nominal playback rate).

If the Media Playlist contains the EXT-X-MEDIA-SEQUENCE tag, the client SHOULD assume that each Media Segment in it will become unavailable at the time that the Playlist file was loaded plus the duration of the Playlist file.

A client MAY use the segment Media Sequence Number to track the location of a Media Segment within a Playlist when the Playlist is reloaded.

A client MUST NOT assume that segments with the same Media Sequence Number in different Variant Streams or Renditions have the same position in the presentation; Playlists MAY have independent Media Sequence Numbers. Instead, a client MUST use the relative position of each segment on the Playlist timeline and its Discontinuity Sequence Number to locate corresponding segments.

Clients using Delivery Directives (Section 6.2.5) MUST ensure that all query parameters appear in UTF-8 order within the URI. This improves Server cache utilization.

A client MUST load the Media Playlist file of every Rendition selected for playback in order to locate the media specific to that Rendition. But, to prevent unnecessary load on the server, it SHOULD NOT load the Playlist file of any other Rendition.

For some Variant Streams, it is possible to select Renditions that do not include the Rendition specified by the EXT-X-STREAM-INF tag. As noted above, the client SHOULD NOT load that Rendition in those cases.

6.3.3. Playing the Media Playlist File

The client SHALL choose which Media Segment to play first from the Media Playlist when playback starts. If the EXT-X-ENDLIST tag is not present and the client intends to play the media normally, the client SHOULD NOT choose a segment closer to the end of the Playlist than described by the HOLD-BACK and PART-HOLD-BACK attributes. Doing so can trigger playback stalls.

Normal playback can be achieved by playing Media Segments or Partial Segments in the order that they appear in the Playlist. The client MAY present the available media in any way it wishes, including normal playback, random access, and trick modes.

The client SHOULD NOT attempt to load Media Segments that have been marked with an EXT-X-GAP tag, or to load Partial Segments with a GAP=YES attribute. Instead, clients are encouraged to look for another Variant Stream of the same Rendition which does not have the same gap, and play that instead.

The encoding parameters for samples in a Media Segment and across multiple Media Segments in a Media Playlist SHOULD remain consistent. However, clients SHOULD deal with encoding changes as they are encountered, for example, by scaling video content to accommodate a resolution change. If the Variant Stream includes a RESOLUTION attribute, clients SHOULD display all video within a rectangle with the same proportions as that resolution.

Clients SHOULD be prepared to handle multiple tracks of a particular type (e.g., audio or video). A client with no other preference SHOULD choose the track with the lowest numerical track identifier that it can play.

Clients SHOULD ignore private streams inside Transport Streams that they do not recognize. Private streams can be used to support different devices with the same stream, although stream authors SHOULD be sensitive to the additional network load that this imposes.

The client MUST be prepared to reset its parser(s) and decoder(s) before playing a Media Segment that has an EXT-X-DISCONTINUITY tag applied to it; otherwise, playback errors can occur.

The client SHOULD attempt to load Media Segments in advance of when they will be required for uninterrupted playback to compensate for temporary variations in latency and throughput.

The client MAY use the value of the EXT-X-PROGRAM-DATE-TIME tag to display the program origination time to the user. If the value includes time zone information, the client SHALL take it into account; if it does not, the client MAY assume the time to be local.

Note that dates in Playlists can refer to when the content was produced (or to other times), which have no relation to the time of playback.

If the first EXT-X-PROGRAM-DATE-TIME tag in a Playlist appears after one or more Media Segment URIs, the client SHOULD extrapolate backward from that tag (using EXTINF durations and/or media timestamps) to associate dates with those segments. To associate a date with any other Media Segment that does not have an EXT-X-PROGRAM-DATE-TIME tag applied to it directly, the client SHOULD extrapolate forward from the last EXT-X-PROGRAM-DATE-TIME tag appearing before that segment in the Playlist.

6.3.4. Reloading the Media Playlist File

The client MUST periodically reload a Media Playlist file to learn what media is currently available, unless it contains an EXT-X-PLAYLIST-TYPE tag with a value of VOD, or a value of EVENT and the EXT-X-ENDLIST tag is also present.

However, the client MUST NOT attempt to reload the Playlist file more frequently than specified by this section, in order to limit the collective load on the server.

When a client loads a Playlist file for the first time or reloads a Playlist file and finds that it has changed since the last time it was loaded, the client MUST wait for at least the duration of the last segment in the Playlist before attempting to reload the Playlist file again, measured from the last time the client began loading the Playlist file.

If the client reloads a Playlist file and finds that it has not changed, then it MUST wait for a period of one-half the Target Duration before retrying. If the Playlist file remains unchanged when reloaded and it has been at least 1.5 times the Target Duration since the last time the client loaded a changed Playlist then the client MAY conclude that the server is not behaving properly and switch to a different Variant Stream or trigger a playback error.

After reloading a Media Playlist, the client SHOULD verify that each Media Segment in it has the same URI (and byte range, if specified) as the Media Segment with the same Media Sequence Number in the previous Media Playlist. It SHOULD halt playback if it does not, as this normally indicates a server error.

In order to reduce server load, the client SHOULD NOT reload the Playlist files of Variant Streams or alternate Renditions that are not currently being played. If it decides to switch playback to a different Variant Stream, it SHOULD stop reloading the Playlist of the old Variant Stream and begin loading the Playlist of the new Variant Stream. It can use the EXTINF durations and the constraints in Section 6.2.4 to determine the approximate location of corresponding media. Once media from the new Variant Stream has been loaded, the timestamps in the Media Segments can be used to synchronize the old and new timelines precisely.

A client MUST NOT attempt to use the Media Sequence Number to synchronize between streams (see Section 6.3.2).

6.3.5. Determining the Next Segment to Load

The client MUST examine the Media Playlist file every time it is loaded or reloaded to determine the next Media Segment to load, as the set of available media MAY have changed.

The first segment to load is generally the segment that the client has chosen to play first (see Section 6.3.3).

In order to play the presentation normally, the next Media Segment to load is the one with the lowest Media Sequence Number that is greater than the Media Sequence Number of the last Media Segment loaded.

6.3.6. Decrypting Encrypted Media Segments

If a Media Playlist file contains an EXT-X-KEY tag that specifies a Key file URI, the client can obtain that Key file and use the key inside it to decrypt all Media Segments to which that EXT-X-KEY tag applies.

A client MUST ignore any EXT-X-KEY tag with an unsupported or unrecognized KEYFORMAT attribute, to allow for cross-device addressability. If the Playlist contains a Media Segment to which only EXT-X-KEY tags with unrecognized or unsupported KEYFORMAT attributes are applied, playback SHOULD fail.

A client MUST NOT attempt to decrypt any segments whose EXT-X-KEY tag has a METHOD attribute that it does not recognize.

If the encryption METHOD is AES-128, AES-128 CBC decryption SHALL be applied to individual Media Segments, whose encryption format is described in Section 4.4.4.4.

If the encryption METHOD is AES-128 and the Media Segment is part of an I-frame Playlist (Section 4.4.3.6) and it has an EXT-X-BYTERANGE tag applied to it, special care needs to be taken in loading and decrypting the segment, because the resource identified by the URI is encrypted in 16-byte blocks from the start of the resource.

The decrypted I-frame can be recovered by first widening its byte range, as specified by the EXT-X-BYTERANGE tag, so that it starts and ends on 16-byte boundaries from the start of the resource.

Next, the byte range is widened further to include a 16-byte block at the beginning of the range. This 16-byte block allows the correct IV for the following block to be calculated.

The widened byte range can then be loaded and decrypted with AES-128 CBC using an arbitrary IV. The number of bytes added to the beginning and the end of the original byte range are discarded from the decrypted bytes; what remains is the decrypted I-frame.

If the encryption METHOD indicates Sample Encryption, decryption SHALL be applied to encrypted media samples within the Media Segment.

An EXT-X-KEY tag with a METHOD of NONE indicates that the Media Segments it applies to are not encrypted.

6.3.7. Requesting Playlist Delta Updates

If a Media Playlist file contains an EXT-X-SERVER-CONTROL tag with a CAN-SKIP-UNTIL attribute and no EXT-X-ENDLIST tag, a Client MAY use the _HLS_skip Delivery Directive to request Playlist Delta Updates.

A Client SHOULD NOT request a Playlist Delta Update unless it already has a version of the Playlist that is no older than one-half of the Skip Boundary.

The client can request a Playlist Delta Update that skips older Media Segments by adding an "_HLS_skip=YES" directive to the Media Playlist URI when it requests the Playlist.

Alternately, if the EXT-X-SERVER-CONTROL tag contains a CAN-SKIP-DATERANGES=YES attribute, the client can request a Playlist Delta Update that skips both older Segments and older EXT-X-DATERANGE tags by adding an "_HLS_skip=v2" directive to the Media Playlist URI when it requests the Playlist.

A Client MUST merge the contents of a Playlist Delta Update with its previous version of the Playlist to form an up-to-date version of the Playlist. If a Client receives a Playlist containing an EXT-X-SKIP tag and finds 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 directive.

6.3.8. Issuing Blocking Requests

Clients MUST NOT request Blocking Playlist Reloads unless the Playlist contains an EXT-X-SERVER-CONTROL tag with a CAN-BLOCK-RELOAD=YES attribute.

If Blocking Playlist Reloads are supported, Clients SHOULD use the _HLS_msn Delivery Directive (and _HLS_part, if the Playlist contains Partial Segments) to obtain Playlist updates in preference to the polling regime described in Section 6.3.4.

If up-to-date information on the next expected Media Sequence Number of a Rendition is not available, a Client SHOULD use a tune-in algorithm such as the one described in Appendix C to obtain a recent version of the Playlist.

Clients MUST ignore EXT-X-PRELOAD-HINT tags with unrecognized TYPE attributes. Clients SHOULD ignore all but the first EXT-X-PRELOAD-HINT tag in a Playlist with a particular TYPE attribute.

When processing a Playlist containing an EXT-X-PRELOAD-HINT tag with TYPE=PART, a Client with sufficient space in its download pipeline that is not already loading the hinted resource SHOULD request it. This will typically happen at the same time as its blocking request for the next Playlist update.

When processing a Playlist containing an EXT-X-PRELOAD-HINT tag with TYPE=MAP, a Client with sufficient space in its download pipeline that has not already cached the hinted Media Initialization Section SHOULD request it.

A Client SHOULD cancel a request for a hinted resource if it is not present in a subsequent Playlist update, such as in an EXT-X-PRELOAD-HINT tag or as part of another tag such as EXT-X-PART. The client SHOULD ignore the results of such requests.

A Client SHOULD recognize when a Partial Segment indicated by an EXT-X-PART tag is a subrange of a hint download and obtain the Partial Segment from the hint download. Clients SHOULD recognize contiguous ranges between existing Partial Segments and Partial Segment hints and avoid duplicate downloads.

7. Protocol Version Compatibility

Protocol compatibility is specified by the EXT-X-VERSION tag. A Playlist that contains tags or attributes that are not compatible with protocol version 1 MUST include an EXT-X-VERSION tag.

A client MUST NOT attempt playback if it does not support the protocol version specified by the EXT-X-VERSION tag, or unintended behavior could occur.

A Media Playlist MUST indicate an EXT-X-VERSION of 2 or higher if it contains:

- The IV attribute of the EXT-X-KEY tag.

A Media Playlist MUST indicate an EXT-X-VERSION of 3 or higher if it contains:

- Floating-point EXTINF duration values.

A Media Playlist MUST indicate an EXT-X-VERSION of 4 or higher if it contains:

- The EXT-X-BYTERANGE tag.
- The EXT-X-I-FRAMES-ONLY tag.

A Media Playlist MUST indicate an EXT-X-VERSION of 5 or higher if it contains:

- An EXT-X-KEY tag with a METHOD of SAMPLE-AES.
- The KEYFORMAT and KEYFORMATVERSIONS attributes of the EXT-X-KEY tag.
- The EXT-X-MAP tag.

A Media Playlist MUST indicate an EXT-X-VERSION of 6 or higher if it contains:

- The EXT-X-MAP tag in a Media Playlist that does not contain EXT-X-I-FRAMES-ONLY.

Note that in protocol version 6, the semantics of the EXT-X-TARGETDURATION tag changed slightly. In protocol version 5 and earlier it indicated the maximum segment duration; in protocol version 6 and later it indicates the the maximum segment duration rounded to the nearest integer number of seconds.

A Master Playlist MUST indicate an EXT-X-VERSION of 7 or higher if it contains:

- "SERVICE" values for the INSTREAM-ID attribute of the EXT-X-MEDIA tag.

A Playlist MUST indicate an EXT-X-VERSION of 8 or higher if it contains:

- Variable substitution.

A Playlist MUST indicate an EXT-X-VERSION of 9 or higher if it contains:

- The EXT-X-SKIP tag.

A Playlist MUST indicate an EXT-X-VERSION of 10 or higher if it contains:

- An EXT-X-SKIP tag that replaces EXT-X-DATERANGE tags in a Playlist Delta Update.

The EXT-X-MEDIA tag and the AUDIO, VIDEO, and SUBTITLES attributes of the EXT-X-STREAM-INF tag are backward compatible to protocol version 1, but playback on older clients may not be desirable. A server MAY consider indicating an EXT-X-VERSION of 4 or higher in the Master Playlist but is not required to do so.

The PROGRAM-ID attribute of the EXT-X-STREAM-INF and the EXT-X-I-FRAME-STREAM-INF tags was removed in protocol version 6.

The EXT-X-ALLOW-CACHE tag was removed in protocol version 7.

8. Playlist Examples

8.1. Simple Media Playlist

```
#EXTM3U
#EXT-X-TARGETDURATION:10
#EXT-X-VERSION:3
#EXTINF:9.009,
http://media.example.com/first.ts
#EXTINF:9.009,
http://media.example.com/second.ts
#EXTINF:3.003,
http://media.example.com/third.ts
#EXT-X-ENDLIST
```

8.2. Live Media Playlist Using HTTPS

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:8
#EXT-X-MEDIA-SEQUENCE:2680

#EXTINF:7.975,
https://priv.example.com/fileSequence2680.ts
#EXTINF:7.941,
https://priv.example.com/fileSequence2681.ts
#EXTINF:7.975,
https://priv.example.com/fileSequence2682.ts
```

8.3. Playlist with Encrypted Media Segments

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-MEDIA-SEQUENCE:7794
#EXT-X-TARGETDURATION:15

#EXT-X-KEY:METHOD=AES-128,URI="https://priv.example.com/key.php?r=52"

#EXTINF:2.833,
http://media.example.com/fileSequence52-A.ts
#EXTINF:15.0,
http://media.example.com/fileSequence52-B.ts
#EXTINF:13.333,
http://media.example.com/fileSequence52-C.ts

#EXT-X-KEY:METHOD=AES-128,URI="https://priv.example.com/key.php?r=53"

#EXTINF:15.0,
http://media.example.com/fileSequence53-A.ts
```

8.4. Master Playlist

```
#EXTM3U
#EXT-X-STREAM-INF:BANDWIDTH=1280000,AVERAGE-BANDWIDTH=1000000
http://example.com/low.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=2560000,AVERAGE-BANDWIDTH=2000000
http://example.com/mid.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=7680000,AVERAGE-BANDWIDTH=6000000
http://example.com/hi.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=65000,CODECS="mp4a.40.5"
http://example.com/audio-only.m3u8
```

8.5. Master Playlist with I-Frames

```
#EXTM3U
#EXT-X-STREAM-INF:BANDWIDTH=1280000
low/audio-video.m3u8
#EXT-X-I-FRAME-STREAM-INF:BANDWIDTH=86000,URI="low/iframe.m3u8"
#EXT-X-STREAM-INF:BANDWIDTH=2560000
mid/audio-video.m3u8
#EXT-X-I-FRAME-STREAM-INF:BANDWIDTH=150000,URI="mid/iframe.m3u8"
#EXT-X-STREAM-INF:BANDWIDTH=7680000
hi/audio-video.m3u8
#EXT-X-I-FRAME-STREAM-INF:BANDWIDTH=550000,URI="hi/iframe.m3u8"
#EXT-X-STREAM-INF:BANDWIDTH=65000,CODECS="mp4a.40.5"
audio-only.m3u8
```

8.6. Master Playlist with Alternative Audio

In this example, the CODECS attributes have been condensed for space. A '\' is used to indicate that the tag continues on the following line with whitespace removed:

```
#EXTM3U
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="aac",NAME="English", \
   DEFAULT=YES,AUTOSELECT=YES,LANGUAGE="en", \
   URI="main/english-audio.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="aac",NAME="Deutsch", \
   DEFAULT=NO,AUTOSELECT=YES,LANGUAGE="de", \
   URI="main/german-audio.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="aac",NAME="Commentary", \
   DEFAULT=NO,AUTOSELECT=NO,LANGUAGE="en", \
   URI="commentary/audio-only.m3u8"
#EXT-X-STREAM-INF:BANDWIDTH=1280000,CODECS="...",AUDIO="aac"
low/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=2560000,CODECS="...",AUDIO="aac"
mid/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=7680000,CODECS="...",AUDIO="aac"
hi/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=65000,CODECS="mp4a.40.5",AUDIO="aac"
main/english-audio.m3u8
```

8.7. Master Playlist with Alternative Video

This example shows three different video Renditions (Main, Centerfield, and Dugout) and three different Variant Streams (low, mid, and high). In this example, clients that did not support the EXT-X-MEDIA tag and the VIDEO attribute of the EXT-X-STREAM-INF tag would only be able to play the video Rendition "Main".

Since the EXT-X-STREAM-INF tag has no AUDIO attribute, all video Renditions would be required to contain the audio.

In this example, the CODECS attributes have been condensed for space. A '\' is used to indicate that the tag continues on the following line with whitespace removed:

```
#EXTM3U
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="low",NAME="Main", \
   DEFAULT=YES,URI="low/main/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="low",NAME="Centerfield", \
   DEFAULT=NO,URI="low/centerfield/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="low",NAME="Dugout", \
   DEFAULT=NO,URI="low/dugout/audio-video.m3u8"

#EXT-X-STREAM-INF:BANDWIDTH=1280000,CODECS="...",VIDEO="low"
low/main/audio-video.m3u8

#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="mid",NAME="Main", \
   DEFAULT=YES,URI="mid/main/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="mid",NAME="Centerfield", \
   DEFAULT=NO,URI="mid/centerfield/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="mid",NAME="Dugout", \
   DEFAULT=NO,URI="mid/dugout/audio-video.m3u8"

#EXT-X-STREAM-INF:BANDWIDTH=2560000,CODECS="...",VIDEO="mid"
mid/main/audio-video.m3u8

#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="hi",NAME="Main", \
   DEFAULT=YES,URI="hi/main/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="hi",NAME="Centerfield", \
   DEFAULT=NO,URI="hi/centerfield/audio-video.m3u8"
#EXT-X-MEDIA:TYPE=VIDEO,GROUP-ID="hi",NAME="Dugout", \
   DEFAULT=NO,URI="hi/dugout/audio-video.m3u8"

#EXT-X-STREAM-INF:BANDWIDTH=7680000,CODECS="...",VIDEO="hi"
hi/main/audio-video.m3u8
```

8.8. Session Data in a Master Playlist

In this example, only the EXT-X-SESSION-DATA is shown:

```
#EXT-X-SESSION-DATA:DATA-ID="com.example.lyrics",URI="lyrics.json"

#EXT-X-SESSION-DATA:DATA-ID="com.example.title",LANGUAGE="en", \
        VALUE="This is an example"
#EXT-X-SESSION-DATA:DATA-ID="com.example.title",LANGUAGE="es", \
        VALUE="Este es un ejemplo"
```

8.9. CHARACTERISTICS Attribute Containing Multiple Characteristics

Certain characteristics are valid in combination, as in:

```
CHARACTERISTICS=
"public.accessibility.transcribes-spoken-dialog,public.easy-to-read"
```

8.10. EXT-X-DATERANGE Carrying SCTE-35 Tags

This example shows two EXT-X-DATERANGE tags that describe a single Date Range, with an SCTE-35 "out" splice_insert() command that is subsequently updated with an SCTE-35 "in" splice_insert() command.

```
#EXTM3U
...
#EXT-X-DATERANGE:ID="splice-6FFFFFF0",START-DATE="2014-03-05T11:
15:00Z",PLANNED-DURATION=59.993,SCTE35-OUT=0xFC002F0000000000FF0
00014056FFFFFF000E011622DCAFF000052636200000000000A0008029896F50
000008700000000

... Media Segment declarations for 60s worth of media

#EXT-X-DATERANGE:ID="splice-6FFFFFF0",DURATION=59.993,SCTE35-IN=
0xFC002A0000000000FF00000F056FFFFFF000401162802E6100000000000A00
08029896F50000008700000000
...
```

8.11. Low-Latency Playlist

This example shows the end of a Playlist that contains Partial Segments. Note that EXT-X-PART tags have been removed from earlier Parent Segments. The Playlist also includes a Preload Hint, a Rendition Report, and a mid-roll advertisement.

```
#EXTM3U
#EXT-X-TARGETDURATION:4
...
#EXTINF:4.00008,
fileSequence268.mp4
#EXTINF:4.00008,
fileSequence269.mp4
#EXTINF:4.00008,
fileSequence270.mp4
#EXT-X-PART:DURATION=2.00004,INDEPENDENT=YES,URI="filePart271.0.mp4"
#EXT-X-PART:DURATION=2.00004,URI="filePart271.1.mp4"
#EXTINF:4.00008,
fileSequence271.mp4
#EXT-X-PART:DURATION=2.00004,INDEPENDENT=YES,URI="filePart272.0.mp4"
#EXT-X-PART:DURATION=0.50001,URI="filePart272.1.mp4"
#EXTINF:2.50005,
fileSequence272.mp4
#EXT-X-DISCONTINUITY
#EXT-X-PART:DURATION=2.00004,INDEPENDENT=YES,URI="midRoll273.0.mp4"
#EXT-X-PART:DURATION=2.00004,URI="midRoll273.1.mp4"
#EXTINF:4.00008,
midRoll273.mp4
#EXT-X-PART:DURATION=2.00004,INDEPENDENT=YES,URI="midRoll274.0.mp4"
#EXT-X-PRELOAD-HINT:TYPE=PART,URI="midRoll274.1.mp4"
#EXT-X-RENDITION-REPORT:URI="/1M/LL-HLS.m3u8",LAST-MSN=274,LAST-PART=1
```

9. Contributors

Significant contributions to the design of this protocol were made by Jim Batson, David Biderman, Bill May, Roger Pantos, Alan Tseng, and Eryk Vershen. Stuart Cheshire helped edit the specification.

Significant contributions to the update of this protocol were made by Bill May, Eryk Vershen, and Peng Zhou.

In particular, Bill May co-authored the first edition of HTTP Live Streaming, [RFC8216], and continues to provide valuable guidance and input.

10. IANA Considerations

IANA has registered the following media type [RFC2046]:

Type name: application

Subtype name: vnd.apple.mpegurl

Required parameters: none

Optional parameters: none

Encoding considerations: encoded as UTF-8, which is 8-bit text. This media type may require encoding on transports not capable of handling 8-bit text. See Section 4 for more information.

Security considerations: See Section 11.

Compression: this media type does not employ compression.

Interoperability considerations: There are no byte-ordering issues, since files are 8-bit text. Applications could encounter unrecognized tags, which SHOULD be ignored.

Published specification: see Section 4.

Applications that use this media type: Multimedia applications such as the iPhone media player in iOS 3.0 and later and QuickTime Player in Mac OS X version 10.6 and later.

Fragment identifier considerations: no Fragment Identifiers are defined for this media type.

Query parameter considerations: the definition of all query parameters for resources of this media type which begin with the string "_HLS_" are reserved by this specification. Currently-defined query parameters are specified in Section 6.2.5.

Additional information:

Deprecated alias names for this type: none
Magic number(s): #EXTM3U
File extension(s): .m3u8, .m3u (see Section 4)
Macintosh file type code(s): none

Person & email address to contact for further information: David Singer, singer AT apple.com.

Intended usage: LIMITED USE

Restrictions on usage: none

Author: Roger Pantos

Change Controller: David Singer

11. Security Considerations

Since the protocol generally uses HTTP to transfer data, most of the same security considerations apply. See Section 15 of HTTP [RFC7230].

Media file parsers are typically subject to "fuzzing" attacks. Implementors SHOULD pay particular attention to code that will parse data received from a server and ensure that all possible inputs are handled correctly.

Playlist files contain URIs, which clients will use to make network requests of arbitrary entities. Clients SHOULD range-check responses to prevent buffer overflows. See also the Security Considerations section of "Uniform Resource Identifier (URI): Generic Syntax" [RFC3986].

Apart from URI resolution, this format does not employ any form of active content.

Clients SHOULD limit each playback session to a reasonable number of concurrent downloads (for example, four) to avoid contributing to denial-of-service attacks.

HTTP requests often include session state ("cookies"), which may contain private user data. Implementations MUST follow cookie restriction and expiry rules specified by "HTTP State Management Mechanism" [RFC6265] to protect themselves from attack. See also the Security Considerations section of that document, and "Use of HTTP State Management" [RFC2964].

Encryption keys are specified by URI. The delivery of these keys SHOULD be secured by a mechanism such as HTTP Over TLS [RFC2818] (formerly SSL) in conjunction with a secure realm or a session token.

12. References

12.1. Normative References

[AC_3] Advanced Television Systems Committee, "Digital Audio Compression (AC-3) (E-AC-3)", ATSC Standard A/52:2010, November 2010, <http://atsc.org/wp-content/uploads/2015/03/A52-201212-17.pdf>.

[AES_128] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", FIPS PUB 197, DOI 10.6028/NIST.FIPS.197, November 2001, <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf>.

[CEA608] Consumer Technology Association, "Line 21 Data Services", ANSI/CTA Standard 608-E, April 2008, <https://standards.cta.tech/kwspub/published_docs/ANSI-CTA-608-E-R-2014-Preview.pdf>.

[CEA708] Consumer Technology Association, "Digital Television (DTV) Closed Captioning", ANSI/CTA Standard CEA-708-E, August 2013, <https://standards.cta.tech/kwspub/published_docs/ANSI-CTA-708-E-Preview.pdf>.

[CICP] International Organization for Standardization, "Information technology - MPEG systems technologies - Part 8: Coding-independent code points", ISO/IEC International Standard 23001-8:2016, 2016, <https://www.iso.org/obp/ui/#iso:std:iso-iec:23001:-8:ed-2:v1:en>.

[CMAF] International Organization for Standardization, "Information technology -- Multimedia application format (MPEG-A) -- Part 19: Common media application format (CMAF) for segmented media", ISO/IEC International Standard 23000-19:2017, December 2017, <https://www.iso.org/standard/71975.html>.

[COMMON_ENC] International Organization for Standardization, "Information technology -- MPEG systems technologies -- Part 7: Common encryption in ISO base media file format files", ISO/IEC International Standard 23001-7:2016, February 2016, <http://www.iso.org/iso/catalogue_detail.htm?csnumber=68042>.

[H_264] International Telecommunications Union, "Advanced video coding for generic audiovisual services", January 2012, <http://www.itu.int/rec/T-REC-H.264>.

[HDCP] Digital Content Protection LLC, "High-bandwidth Digital Content Protection System - Mapping HDCP to HDMI", February 2013, <http://www.digital-cp.com/sites/default/files/specifications/HDCP%20on%20HDMI%20Specification%20Rev2_2_Final1.pdf>.

[IMSC1] W3C, "TTML Profiles for Internet Media Subtitles and Captions 1.0 (IMSC1)", April 2016, <https://www.w3.org/TR/ttml-imsc1/>.

[ISO_13818]
International Organization for Standardization, "Generic coding of moving pictures and associated audio information", ISO/IEC International Standard 13818:2007, October 2007, <http://www.iso.org/iso/catalogue_detail?csnumber=44169>.

[ISO_13818_3]
International Organization for Standardization, "Generic coding of moving pictures and associated audio information -- Part 3: Audio", ISO/IEC International Standard 13818-3:1998, April 1998, <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=26797>.

[ISO_13818_7]
International Organization for Standardization, "Generic coding of moving pictures and associated audio information -- Part 7: Advanced Audio Coding (AAC)", ISO/IEC International Standard 13818-7:2006, January 2006, <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43345>.

[ISO_14496]
International Organization for Standardization, "Information technology -- Coding of audio-visual objects -- Part 3: Audio", ISO/IEC International Standard 14496-3:2009, 2009, <http://www.iso.org/iso/catalogue_detail?csnumber=53943>.

[ISO_8601]
International Organization for Standardization, "Data elements and interchange formats -- Information interchange -- Representation of dates and times", ISO/IEC International Standard 8601:2004, December 2004, <http://www.iso.org/iso/catalogue_detail?csnumber=40874>.

[ISOBMFF] International Organization for Standardization, "Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format", ISO/IEC International Standard 14496-12:2015, December 2015, <http://www.iso.org/iso/catalogue_detail.htm?csnumber=68960>.

[MP4_TIMED_TEXT]
International Organization for Standardization, "Information technology -- Coding of audio-visual objects -- Part 30: Timed text and other visual overlays in ISO base media file format", ISO/IEC International Standard 14496-30:2014, March 2014, <https://www.iso.org/standard/63107.html>.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

[RFC2964] Moore, K. and N. Freed, "Use of HTTP State Management", BCP 44, RFC 2964, DOI 10.17487/RFC2964, October 2000, <https://www.rfc-editor.org/info/rfc2964>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, DOI 10.17487/RFC6265, April 2011, <https://www.rfc-editor.org/info/rfc6265>.

[RFC6381] Gellens, R., Singer, D., and P. Frojdh, "The ’Codecs’ and ’Profiles’ Parameters for "Bucket" Media Types", RFC 6381, DOI 10.17487/RFC6381, August 2011, <https://www.rfc-editor.org/info/rfc6381>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8216] Pantos, R., Ed. and W. May, "HTTP Live Streaming", RFC 8216, DOI 10.17487/RFC8216, August 2017, <https://www.rfc-editor.org/info/rfc8216>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8673] Pratt, C., Thakore, D., and B. Stark, "HTTP Random Access and Live Content", RFC 8673, DOI 10.17487/RFC8673, November 2019, <https://www.rfc-editor.org/info/rfc8673>.

[SCTE35] Society of Cable Telecommunications Engineers, "Digital Program Insertion Cueing Message for Cable", ANSI/SCTE 35, August 2014, <http://www.scte.org/documents/pdf/Standards/ANSI_SCTE%2035%202014.pdf>.

[US_ASCII] American National Standards Institute, "Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII)", ANSI X3.4, December 1986.

[WebVTT] World Wide Web Consortium (W3C), ""WebVTT: The Web Video Text Tracks Format", Draft Community Group Report", July 2013, <http://dev.w3.org/html5/webvtt/>.

12.2. Informative References

[ID3] ID3.org, "The ID3 audio file data tagging format", <http://www.id3.org/Developer_Information>.

[M3U] "M3U (MP3 URL)", <http://wikipedia.org/wiki/M3U>.

[MCT] Apple Inc., "QuickTime File Format Specification - Media Characteristic Tags", <https://developer.apple.com/library/archive/documentation/QuickTime/QTFF/QTFFChap2/qtff2.html>.

[SampleEnc] Apple Inc., "MPEG-2 Stream Encryption Format for HTTP Live Streaming", <https://developer.apple.com/library/ios/documentation/AudioVideo/Conceptual/HLS_Sample_Encryption/>.

[UNICODE] The Unicode Consortium, "The Unicode Standard", <https://www.unicode.org/versions/latest/>.

Appendix A. Changes from RFC 8216

Several changes have been made since the publication of RFC 8216 [RFC8216].

The following tags have been added: EXT-X-GAP, EXT-X-BITRATE, EXT-X-SERVER-CONTROL, EXT-X-SKIP, EXT-X-PART-INF, EXT-X-PART, EXT-X-PRELOAD-HINT, and EXT-X-RENDITION-REPORT.

The EXT-X-DEFINE tag was introduced to support variable substitution.

IMSC has been added to the set of recognized subtitle formats.

The VIDEO-RANGE, ALLOWED-CPC, and STABLE-VARIANT-ID attributes have been added to the EXT-X-STREAM-INF and EXT-X-I-FRAME-STREAM-INF tags.

The STABLE-RENDITION-ID attribute has been added to the EXT-X-MEDIA tag.

TYPE-1 has been added as a defined value for the HDCP-LEVEL attribute.

Redefined the CHANNELS attribute value.

The minimum new segment publication latency has been removed from server timing model.

The Availability Duration of a Media Segment now depends on the presence of an EXT-X-ENDLIST tag.

The recommended playlist offset to join a live stream has changed.

The minimum delay before reloading a Playlist file has changed.

The definition of peak segment bit rate was changed to ensure every segment is included in at least one contiguous set.

Media Metadata tags such as EXT-X-DATERANGE may be removed from playlists.

Partial Segments were defined as a means to reduce publishing latency.

Delivery Directives were introduced, including support for Playlist Delta Updates and Blocking Playlist Reload.

A Low-Latency Server Configuration Profile was added to Appendix B.

Correct the reference for the semantics of the CHARACTERISTICS attribute.

Appendix C was added.

There have been a number of minor editorial changes.

Appendix B. Server Configuration Profiles

Server Configuration Profiles specify additional requirements that optimize delivery of HTTP Live Streaming for certain use cases.

B.1. Low-Latency Server Configuration Profile

Playing at a reduced delay from live requires certain stream and transport features to support the timely delivery of media. Clients SHOULD verify that the server meets these requirements before playing at a delay-from-live of less than two Target Durations. Because the Low-Latency extensions are additions rather than replacements, clients can and SHOULD fall back to regular-latency playback if they discover that the server does not meet the requirements of this configuration profile.

This profile places the following requirements on stream production:

All Media Playlists have EXT-X-PROGRAM-DATE-TIME tags. This allows more-precise mapping between Segments across Renditions. Note that real-time clocks are NOT required to be synchronized between client and server.

Each (non-Partial) Media Segment in a Media Playlist will contain at least one independent frame.

A Playlist that contains an EXT-X-PART tag but no EXT-X-ENDLIST tag will also contain an EXT-X-PRELOAD-HINT tag that specifies the next Partial Segment that is expected to be added to the Playlist.

If the Partial Segment specified by an EXT-X-PRELOAD-HINT tag has a different Media Initialization Section than the last Partial Segment in the Playlist, the Playlist will also contain an EXT-X-PRELOAD-HINT tag with TYPE=MAP that hints the Media Initialization Section of the hinted Partial Segment.

Each Media Playlist contains one EXT-X-RENDITION-REPORT tag for each Media Playlist (Rendition) in the Master Playlist, except for the Media Playlist to which the EXT-X-RENDITION-REPORT tag is being added, and Playlists that contain the EXT-X-I-FRAMES-ONLY tag.

This profile places the following requirements on stream delivery:

HTTP-delivered Playlists and Segments are served via HTTP/2. Efficient delivery requires HTTP/2 priority control (dependencies and weights) and support for Ping frames.

Each server offers the entire set of Variant Streams in the master Playlist. This allows rapid bit rate switching without connection reestablishment.

Servers support HTTP Range requests if Media Playlists contain the BYTERANGE, BYTERANGE-START, or BYTERANGE-LENGTH attributes.

TCP connections support Selective Acknowledgment (SACK) across the entire route from client to server. This improves the performance of TCP loss recovery

Playlist requests are idempotent.

Playlists are delivered in GZIP format. This speeds up Media Playlist reload and Rendition switching.

All Renditions in a Master Playlist are updated in sync, within an accuracy of one Part Target Duration.

CDNs and other proxy caches recognize blocking requests for Playlists and Media Segments whose cache fill is already pending, and hold the duplicate requests until they can be delivered from that cache fill. This minimizes the load on the active origin.

HTTP caches used to deliver Playlists or Segments will set the Age HTTP Response header.

In addition, the following configurations are recommended:

TCP connections should set Explicit Congestion Notification (ECN) during congestion. They should also use TCP timestamps, TAIL LOSS probe, and TCP RACK. These configurations 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.

Servers should support TLS 1.3 or higher. This reduces time to connect. Servers should also support TLS 1.3 0-RTT connections for Media Playlists and Media Segments.

Blocking Playlist Reload allows longer caching of Playlists without detriment to Clients. Successful responses to blocking Playlist requests should be cached for six Target Durations. Unsuccessful responses (such as 404s) should be cached for four Target Durations. Successful responses to non-blocking Playlist requests should be cached for half the Target Duration. Unsuccessful responses to non-blocking Playlist requests should be cached for for one Target Duration.

Successful responses to blocking Media segment requests should be cached for six Target Durations. Unsuccessful responses should be cached for one Target Duration.

Origin servers should use Cache-Control headers to communicate the desired cache lifetime.

The recommended Target Duration is six seconds.

The recommended GOP size is between one and two seconds. Smaller GOPs allow faster switching between Renditions.

Appendix C. Low-Latency CDN Tune-in

Clients SHOULD support delivery of low-latency streams through CDNs and other HTTP caches. Correctly implementing PART-HOLD-BACK, the server-recommended playback delay from live, requires that the client first obtain a reasonably up-to-date version of the Media Playlist.

There are various approaches that a client may take to obtain a recent version of a Media Playlist. The following algorithm typically requires two Playlist requests to obtain a Playlist that is within one Part Target Duration of the current Playlist:

1. Send a request for the Media Playlist that does not include an _HLS_msn or _HLS_part directive.

2. Record the first Playlist response, including its received time and Age header. If there's no Age header in the first Playlist response, consider the Playlist to be up to date. (No Age header means that the response came directly from the origin, rather than being held for a period of time in an intervening HTTP cache.)

3. If there is an Age header in the first Playlist response, set the goalDuration to match the Age value. Increase the goalDuration by one second if the Part Target Duration is less than 1.0.

4. While the Age value is greater than or equal to the floor of the Part Target Duration:

    A. Set currentGoal to be the goalDuration plus the amount of time since the first Playlist response.

    B. If the current version of the Playlist has at least currentGoal more media in it than the first Playlist, consider the current Playlist to be up to date.

    C. Use the Target Duration and the Part Target Duration to estimate how many more segments and parts the server will add to the Playlist to contribute at least currentGoal more media to it.

    D. Request the Media Playlist again, using the _HLS_msn and _HLS_part directives to obtain the Playlist that has the estimated additional duration of media since the first Playlist.

    E. Update the current Playlist and the Age value from the Playlist response.

Author's Address

Roger Pantos (editor)
Apple Inc.
Cupertino, California
United States

Email: http-live-streaming-review@group.apple.com