

Informational
Internet-Draft
Intended status: Informational
Expires: October 26, 2017

R. Pantos, Ed.
Apple Inc.
W. May
Major League Baseball Advanced Media
April 24, 2017

HTTP Live Streaming draft-pantos-http-live-streaming-22

Abstract

This document 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 7 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 <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on October 26, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

This document may not be modified, and derivative works of it may not

be created, and it may not be published except as an Internet-Draft.

This Informational Internet Draft is submitted as an RFC Editor Contribution and/or non-IETF Document (not as a Contribution, IETF Contribution, nor IETF Document) in accordance with [BCP 78](#) and [BCP 79](#).

Table of Contents

1.	Introduction to HTTP Live Streaming	3
2.	Overview	4
3.	Media Segments	5
3.1.	Supported Media Segment Formats	6
3.2.	MPEG-2 Transport Streams	6
3.3.	Fragmented MPEG-4	6
3.4.	Packed Audio	7
3.5.	WebVTT	8
4.	Playlists	8
4.1.	Definition of a Playlist	9
4.2.	Attribute Lists	10
4.3.	Playlist Tags	11
4.3.1.	Basic Tags	11
4.3.1.1.	EXTM3U	11
4.3.1.2.	EXT-X-VERSION	12
4.3.2.	Media Segment Tags	12
4.3.2.1.	EXTINF	12
4.3.2.2.	EXT-X-BYTERANGE	13
4.3.2.3.	EXT-X-DISCONTINUITY	13
4.3.2.4.	EXT-X-KEY	14
4.3.2.5.	EXT-X-MAP	16
4.3.2.6.	EXT-X-PROGRAM-DATE-TIME	17
4.3.2.7.	EXT-X-DATERANGE	17
4.3.2.7.1.	Mapping SCTE-35 into EXT-X-DATERANGE	19
4.3.3.	Media Playlist Tags	21
4.3.3.1.	EXT-X-TARGETDURATION	21
4.3.3.2.	EXT-X-MEDIA-SEQUENCE	21
4.3.3.3.	EXT-X-DISCONTINUITY-SEQUENCE	22
4.3.3.4.	EXT-X-ENDLIST	22
4.3.3.5.	EXT-X-PLAYLIST-TYPE	23
4.3.3.6.	EXT-X-I-FRAMES-ONLY	23
4.3.4.	Master Playlist Tags	24
4.3.4.1.	EXT-X-MEDIA	24
4.3.4.1.1.	Rendition Groups	27
4.3.4.2.	EXT-X-STREAM-INF	28
4.3.4.2.1.	Alternative Renditions	31
4.3.4.3.	EXT-X-I-FRAME-STREAM-INF	32
4.3.4.4.	EXT-X-SESSION-DATA	33
4.3.4.5.	EXT-X-SESSION-KEY	34

4.3.5.	Media or Master Playlist Tags	34
4.3.5.1.	EXT-X-INDEPENDENT-SEGMENTS	34
4.3.5.2.	EXT-X-START	35
5.	Key files	36
5.1.	Structure of Key files	36
5.2.	IV for [AES 128]	36
6.	Client/Server Responsibilities	36
6.1.	Introduction	36
6.2.	Server Responsibilities	36
6.2.1.	General Server Responsibilities	36
6.2.2.	Live Playlists	39
6.2.3.	Encrypting Media Segments	40
6.2.4.	Providing Variant Streams	41
6.3.	Client Responsibilities	42
6.3.1.	General Client Responsibilities	42
6.3.2.	Loading the Media Playlist file	43
6.3.3.	Playing the Media Playlist file	44
6.3.4.	Reloading the Media Playlist file	45
6.3.5.	Determining the next segment to load	46
6.3.6.	Decrypting encrypted Media Segments	46
7.	Protocol version compatibility	47
8.	Playlist Examples	48
8.1.	Simple Media Playlist	48
8.2.	Live Media Playlist, using HTTPS	48
8.3.	Playlist with encrypted Media Segments	49
8.4.	Master Playlist	49
8.5.	Master Playlist with I-Frames	50
8.6.	Master Playlist with Alternative audio	50
8.7.	Master Playlist with Alternative video	50
8.8.	Session Data in a Master Playlist	51
8.9.	CHARACTERISTICS attribute containing multiple characteristics	52
8.10.	EXT-X-DATERANGE carrying SCTE-35 tags	52
9.	Contributors	52
10.	IANA Considerations	52
11.	Security Considerations	53
12.	References	54
12.1.	Normative References	54
12.2.	Informative References	57
	Authors' Addresses	57

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.

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 eight 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 7 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.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

3. Media Segments

A Media Playlist contains a series of Media Segments which 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 4.3.2.1](#)).

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 4.3.3.2](#)). 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 which are explicitly signaled as discontinuities

([Section 4.3.2.3](#)). 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 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 ([Section 4.3.2.5](#)) tag. The Media Initialization Section MUST NOT contain sample data.

[3.2.](#) 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 ([Section 4.3.2.5](#)) tag 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.

[3.3.](#) Fragmented MPEG-4

MPEG-4 Fragments are specified by the ISO Base Media File Format [[ISOBMFF](#)]. Unlike regular MPEG-4 files which 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](#) [\[ISOBMFF\]](#).

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. A Movie Extends Box ('mvex') MUST follow the last Track Box.

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.

An fMP4 Segment in a Playlist containing the EXT-X-I-FRAMES-ONLY ([Section 4.3.3.6](#)) tag MAY omit the portion of the Media Data Box following the I-frame sample data.

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

[3.4.](#) 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 AAC with 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 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.

3.5. 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.

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, a X-TIMESTAMP-MAP metadata header **SHOULD** be added to each WebVTT header. This header maps WebVTT cue timestamps to MPEG-2 (PES) timestamps in other Renditions of the Variant Stream. Its format is:

```
X-TIMESTAMP-MAP=LOCAL:<cue time>,MPEGTS:<MPEG-2 time>  
e.g. X-TIMESTAMP-MAP=LOCAL:00:00:00.000,MPEGTS:900000
```

The cue timestamp in the LOCAL attribute **MAY** fall outside the range of time covered by the segment.

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 MPEG-2 timestamp of 0.

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 4.3.1.1](#)) and EXTINF ([Section 4.3.2.1](#)).

In this section, tags are specified using a BNF-style syntax.

Each Playlist file MUST be identifiable by either 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 reject Playlists which 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". 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 an line feed character. Each line is a URI, blank, or starts with the character '#'. Blank lines are ignored. White space 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 4.3.4.2](#)). 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 file 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 4.3.2.1](#)). 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 and 1.5 times the target duration. 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 which 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:

- o 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.
- o 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 AttributeName.

- o decimal-floating-point: an unquoted string of characters from the set [0..9] and '.' which expresses a non-negative floating-point number in decimal positional notation.
- o signed-decimal-floating-point: an unquoted string of characters from the set [0..9], '-' and '.' which expresses a signed floating-point number in decimal positional notation.
- o 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). 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.
- o enumerated-string: an unquoted character string from a set which is explicitly defined by the Attribute. An enumerated-string will never contain double-quotes ("), commas (,), or whitespace.
- o 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 play such Playlists.

4.3. Playlist Tags

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

4.3.1. Basic Tags

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

4.3.1.1. 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
```


[4.3.1.2.](#) 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 reject it.

[4.3.2.](#) 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 reject Playlists that contain both Media Segment Tags and Master Playlist tags ([Section 4.3.4](#)).

[4.3.2.1.](#) 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. Generally, durations SHOULD be decimal-floating-point, with enough accuracy to avoid perceptible error when segment durations are accumulated. 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 raw UTF-8 text.

[4.3.2.2](#). 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 Playlist MUST be rejected.

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.

[4.3.2.3](#). 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:

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

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

- o encoding parameters
- o encoding sequence

See [Section 3](#), [Section 6.2.1](#), and [Section 6.3.3](#) for more information about the EXT-X-DISCONTINUITY tag.

[4.3.2.4](#). EXT-X-KEY

Media Segments MAY be encrypted. The EXT-X-KEY tag specifies how to decrypt them. It applies to every Media Segment that appears between it and the next EXT-X-KEY tag in the Playlist file with the same KEYFORMAT attribute (or the end of the Playlist file). 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 128](#)] with a 128-bit key, Cipher Block Chaining, and PKCS7 padding [[RFC5652](#)]. CBC is restarted on each segment boundary, using either the IV attribute value or the Media Sequence Number as the IV; see [Section 5.2](#).

An encryption method of SAMPLE-AES means that the Media Segments contain media samples, such as audio or video, that are 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 HLS Sample Encryption [[SampleEnc](#)] specification. 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.

[4.3.2.5](#). 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 4.3.2.2](#). 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.

4.3.2.6. 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.

The date/time representation is ISO/IEC 8601:2004 [[ISO 8601](#)] and SHOULD indicate a time zone and fractional parts of seconds:

```
#EXT-X-PROGRAM-DATE-TIME:<YYYY-MM-DDThh:mm:ss.SSSZ>
```

For example:

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

EXT-X-PROGRAM-DATE-TIME tags SHOULD provide millisecond accuracy.

See [Section 6.2.1](#) and [Section 6.3.3](#) for more information on the EXT-X-PROGRAM-DATE-TIME tag.

4.3.2.7. 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 attribute-value semantics. This attribute is OPTIONAL.

START-DATE

A quoted-string containing the ISO-8601 date at which the Date Range begins. This attribute is REQUIRED.

END-DATE

A quoted-string containing the ISO-8601 date 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 4.3.2.7.1](#) 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, or 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.

4.3.2.7.1. 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 a SCTE35-CMD attribute whose value is the big-endian binary representation of the splice_info_section(), expressed as a hexadecimal-sequence.

A 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 a 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 a SCTE35-IN attribute, with the same formatting as SCTE35-CMD.

A 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 a SCTE35-OUT attribute; the "in" `splice_info_section()` MUST be placed in a 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 a SCTE35-OUT, SCTE35-IN or SCTE35-CMD attribute whose value is the entire `splice_info_section()`.

A 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 a SCTE35-CMD attribute.

The START-DATE of an EXT-X-DATERANGE tag containing a 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 a SCTE35-CMD MUST be the date and time specified by the `splice_time()` in the command, or to 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 a 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 a 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 a 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 a 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.

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 the Playlist. If there is a possibility that a SCTE-35 id will be reused, the ID attribute value MUST include disambiguation, such as a date or sequence number.

4.3.3. 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

4.3.3.1. EXT-X-TARGETDURATION

The EXT-X-TARGETDURATION tag specifies the maximum Media Segment duration. The EXTINF duration of each Media Segment in the 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.

4.3.3.2. 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.

[4.3.3.3](#). 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.

[4.3.3.4](#). 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
```


4.3.3.5. 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:<EVENT|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 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.

4.3.3.6. 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 (or Intra frames) are encoded video frames whose encoding 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 4.3.2.2](#)) MUST NOT include its Media Initialization Section; clients can assume that the Media Initialization Section is defined

by EXT-X-MAP tag, or is located from the start of the resource to 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.

4.3.4. 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 reject any Playlist that contains both a Master Playlist tag and either a Media Playlist tag or a Media Segment tag.

4.3.4.1. EXT-X-MEDIA

The EXT-X-MEDIA tag is used to relate Media Playlists that contain alternative Renditions ([Section 4.3.4.2.1](#)) 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.

In Transport Streams, Closed captions [[CEA608](#)] media is carried in the video stream. Therefore, an EXT-X-MEDIA tag with TYPE of CLOSED-CAPTION 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 4.3.4.2.1](#). If the TYPE is CLOSED-CAPTIONS, the URI attribute MUST NOT be present.

GROUP-ID

The value is a quoted-string which specifies the group to which the Rendition belongs. See [Section 4.3.4.1.1](#). 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 vs. spoken, or as a fallback dialect). This attribute is OPTIONAL.

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

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.

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 which 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 which 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 "SERVICE n " where n MUST be an integer between 1 and 63 (e.g. "SERVICE3" 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 Uniform Type Identifiers [UTI] separated by comma (,) characters. This attribute is OPTIONAL. Each UTI 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"; "public.easy-to-read" (which indicates that the subtitles have been edited for ease of reading).

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

The CHARACTERISTICS attribute MAY include private UTIs.

CHANNELS

The value is a quoted-string that specifies an ordered, "/"-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. 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 encoded with the same codec but a different number of channels, then the CHANNELS attribute is REQUIRED; otherwise it is OPTIONAL.

4.3.4.1.1. 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:

- o All EXT-X-MEDIA tags in the same Group MUST have different NAME attributes.
- o A Group MUST NOT have more than one member with a DEFAULT attribute of YES.
- o 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 ([Section 4.3.4.2](#)) tag or EXT-X-I-FRAME-STREAM-INF ([Section 4.3.4.3](#)) tag which 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".

4.3.4.2. EXT-X-STREAM-INF

The EXT-X-STREAM-INF tag specifies a Variant Stream, which is a set of Renditions which 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.

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-LC audio and H.264 Main Profile Level 3.0 video would have a CODECS value of "mp4a.40.2,avc1.4d401e".

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 3 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 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 HDCP Type 0 [[HDCP](#)] 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.

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 4.3.4.2.1](#).

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 4.3.4.2.1](#).

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 4.3.4.2.1](#).

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 indicates the set of closed-caption Renditions that can be used when playing the presentation. See [Section 4.3.4.2.1](#).

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.

[4.3.4.2.1](#). 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:

- o 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.
- o 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.
- o 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.

4.3.4.3. 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 4.3.4.2](#)) 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 4.3.4.2.1](#) 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.

4.3.4.4. 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 which identifies that 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 which 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 [[RFC7159](#)]; 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.

4.3.4.5. 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 4.3.2.4](#)) 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. A 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.

4.3.5. 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).

These tags MUST NOT appear more than once in a Playlist. If a tag appears more than once, clients MUST reject the playlist.

4.3.5.1. 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.

4.3.5.2. 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.

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 Initialization Vector (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 Initialization Vector 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 is less than or equal to a constant target duration. Segments that are longer than the planned target duration 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, e.g. 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.

Any Media Segment that is specified in a Playlist loaded by a client MUST be available for immediate download, or playback errors can occur. Once download starts, its transfer rate SHOULD NOT be constrained by the segment production process.

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 4.3.1.2](#)) 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](#)).

- 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 delete any part of the Playlist file; it MAY append lines to it.

The value of the EXT-X-TARGETDURATION tag in the Media Playlist MUST NOT change. A typical target duration is 10 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.

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 relative to the time that the previous version of the Playlist file was made available: no earlier than one-half the target duration after that time, and no later than 1.5 times the target duration after that time. This allows clients to utilize the network efficiently.

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.

Media Segments MUST be removed from the Playlist file 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.

When the server removes a Media Segment URI from the Playlist, the corresponding Media Segment MUST remain available to clients for a period of time equal to the duration of the segment plus the duration of the longest Playlist file distributed by the server containing that segment. Removing a Media Segment earlier than that can interrupt in-progress playback.

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 4.3.2.4](#)) 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.

If an encrypted Media Segment is followed by one or more unencrypted Media Segments in the Playlist, the unencrypted Media Segments MUST have an EXT-X-KEY tag whose METHOD is NONE applied to them. 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 4.3.2.4](#) 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 is SAMPLE-AES, 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 4.3.3.3](#).

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 the 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, each one identified by EXT-X-DATERANGE tag(s) with the same ID attribute value and containing the same set of attributes / value pairs.

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 4.3.4.2.1](#).

[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.

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**:

- o ignore any unrecognized tags.
- o ignore any Attribute/value pair with an unrecognized AttributeName.
- o 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.

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.

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 which starts less than three target durations from the end of the Playlist file. Doing so can trigger playback stalls.

Normal playback can be achieved by playing the Media 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 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 target duration 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.

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 4.3.2.4](#).

If the encryption METHOD is AES-128 and the Media Segment is part of an I-frame playlist ([Section 4.3.3.6](#)) 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 is SAMPLE-AES, AES-128 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.

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 a EXT-X-VERSION of 2 or higher if it contains:

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

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

- o Floating-point EXTINF duration values.

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

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

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

- o The KEYFORMAT and KEYFORMATVERSIONS attributes of the EXT-X-KEY tag.
- o The EXT-X-MAP tag.

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

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

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

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

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 a 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 3 different video Renditions (Main, Centerfield and Dugout), and 3 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 a 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
00014056FFFFFFF000E011622DCAFF0000526362000000000000A0008029896F50
0000087000000000

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

#EXT-X-DATERANGE:ID="splice-6FFFFFF0",DURATION=59.993,SCTE35-IN=
0xFC002A0000000000FF00000F056FFFFFFF000401162802E6100000000000A00
08029896F500000087000000000
...
```

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.

10. IANA Considerations

This memo requests that the following MIME type [[RFC2046](#)] be registered with the IANA:

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.

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)

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

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 URL resolution, this format does not employ any form of active content.

Clients SHOULD limit each playback session to a reasonable number of concurrent downloads (e.g. 4) 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, "ATSC Standard: A/52:2010: Digital Audio Compression (AC-3) (E-AC-3) Standard", November 2010, <<http://atsc.org/wp-content/uploads/2015/03/A52-201212-17.pdf>>.
- [AES_128] U.S. Department of Commerce/National Institute of Standards and Technology, "Advanced Encryption Standard (AES), FIPS PUB 197", November 2001, <<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>>.
- [CEA608] Consumer Electronics Association, "CEA-608-E: Line 21 Data Services", April 2008, <<http://www.ce.org/Standards/Standard-Listings/R4-3-Television-Data-Systems-Subcommittee/Line-21-Data-Service.aspx>>.

[CEA708] Consumer Electronics Association, "CEA-708-E (ANSI): Digital Television (DTV) Closed Captioning", August 2013, <<http://www.ce.org/Standards/Standard-Listings/R4-3-Television-Data-Systems-Subcommittee/CEA-708-D.aspx>>.

[COMMON_ENC]

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

[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>.

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

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

[ISO_13818]

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

[ISO_13818_3]

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

[ISO_13818_7]

International Organization for Standardization, "ISO/IEC International Standard 13818-3:2006; Generic coding of moving pictures and associated audio information - Part 7: Advanced Audio Coding (AAC)", January 2006, <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43345>.

- [ISO_14496] International Organization for Standardization, "ISO/IEC 14496-3:2009 Information technology -- Coding of audio-visual objects -- Part 3: Audio", 2009, <http://www.iso.org/iso/catalogue_detail?csnumber=53943>.
- [ISO_8601] International Organization for Standardization, "ISO/IEC International Standard 8601:2004; Data elements and interchange formats -- Information interchange -- Representation of dates and times", December 2004, <http://www.iso.org/iso/catalogue_detail?csnumber=40874>.
- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", [RFC 2046](#), November 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2818] Rescorla, E., "HTTP Over TLS", [RFC 2818](#), May 2000.
- [RFC2964] Moore, K. and N. Freed, "Use of HTTP State Management", [BCP 44](#), [RFC 2964](#), October 2000.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, [RFC 3629](#), November 2003.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), January 2005.
- [RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", [BCP 47](#), [RFC 5646](#), September 2009.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, [RFC 5652](#), September 2009.
- [RFC6265] Barth, A., "HTTP State Management Mechanism", [RFC 6265](#), April 2011.
- [RFC6381] Gellens, R., Singer, D., and P. Frojdh, "The 'Codecs' and 'Profiles' Parameters for 'Bucket' Media Types", [RFC 6381](#), August 2011.
- [RFC7159] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", [RFC 7159](#), March 2014.

- [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, <<http://www.rfc-editor.org/info/rfc7230>>.
- [SCTE35] Society of Cable Telecommunications Engineers, "SCTE-35 (ANSI/SCTE): Digital Program Insertion Cueing Message for Cable", August 2014, <http://www.scte.org/documents/pdf/Standards/ANSI_SCTE%2035%202014.pdf>.
- [US_ASCII] American National Standards Institute, "ANSI X3.4-1986, Information Systems -- Coded Character Sets 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII)", December 1986.
- [WebVTT] World Wide Web Consortium (W3C), "WebVTT: The Web Video Text Tracks Format", 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] Nullsoft, Inc., "The M3U Playlist format, originally invented for the Winamp media player", <<http://wikipedia.org/wiki/M3U>>.
- [SampleEnc] Apple Inc., "MPEG-2 Stream Encryption Format for HTTP Live Streaming", <https://developer.apple.com/library/ios/documentation/AudioVideo/Conceptual/HLS_Sample_Encryption/>.
- [UTI] Apple Inc., "Uniform Type Identifier", <<http://developer.apple.com/library/ios/#documentation/general/conceptual/DevPedia-CocoaCore/UniformTypeIdentifier.html>>.

Authors' Addresses

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

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

William May, Jr.
Major League Baseball Advanced Media
New York, New York
United States

Email: bill.may@mlb.com