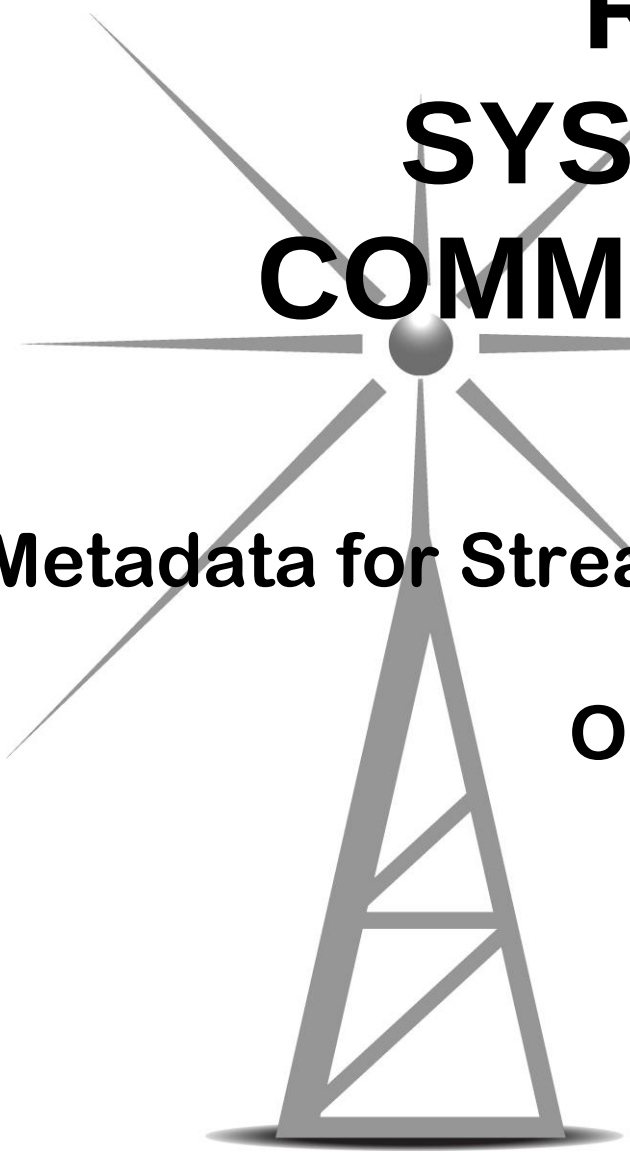
NRSC GUIDELINE

NATIONAL RADIO SYSTEMS COMMITTEE



NRSC-G304
Metadata for Streaming Audio Handbook
October 2023

NAB: 1 M Street SE
Washington, DC 20003
Tel: 202-429-5346
www.nab.org

1919 South Eads Street
Arlington, VA 22202
Tel: 703-907-7652
www.cta.tech

Co-sponsored by the Consumer Technology Association and the National Association of Broadcasters
http://www.nrscstandards.org

NOTICE

NRSC Standards, Guidelines, Reports and other technical publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such Standards, Guidelines, Reports and other technical publications shall not in any respect preclude any member or nonmember of the Consumer Technology Association (CTA) or the National Association of Broadcasters (NAB) from manufacturing or selling products not conforming to such Standards, Guidelines, Reports and other technical publications, nor shall the existence of such Standards, Guidelines, Reports and other technical publications preclude their voluntary use by those other than CTA or NAB members, whether to be used either domestically or internationally.

Standards, Guidelines, Reports and other technical publications are adopted by the NRSC in accordance with the NRSC patent policy. By such action, CTA and NAB do not assume any liability to any patent owner, nor do they assume any obligation whatever to parties adopting the Standard, Guideline, Report or other technical publication.

This Standard does not purport to address all safety problems associated with its use or all applicable regulatory requirements. It is the responsibility of the user of this Standard to establish appropriate safety and health practices and to determine the applicability of regulatory limitations before its use.

Published by
CONSUMER TECHNOLOGY ASSOCIATION
Technology & Standards Department
1919 S. Eads St.
Arlington, VA 22202

NATIONAL ASSOCIATION OF BROADCASTERS
Technology Department
1 M Street SE
Washington, DC 20003

©2023 CTA & NAB. All rights reserved.

This document is available free of charge via the NRSC website at www.nrscstandards.org. Republication or further distribution of this document, in whole or in part, requires prior permission of CTA or NAB.

FOREWORD

Thanks to ubiquitous mobile broadband and smartphones, consumers can access a cornucopia of audio programming. Many radio broadcasters provide audio streaming versions of over-the-air (OTA) radio station content and pure-play streams. Text and image metadata have become an important part of a radio station's OTA and streaming offerings.

This Guideline for radio broadcasters and netcasters describes how to use metadata with audio streams. It focuses on the HTTP live streaming (HLS) and Dynamic Adaptive Streaming over HTTP (MPEG-DASH) methods of audio streaming because these are modern, reliable, efficient, and standards-based. Correctly implementing these modern segmented streaming formats requires a completely different workflow than previous legacy ICY methods (SHOUTcast and Icecast), but pays off with much higher reliability and richer, on-time metadata. There are many additional benefits, listed in section 5.1, to switching from ICY to HLS or MPEG-DASH streaming.

The DSM Subcommittee hopes this document will be useful to streamers and broadcasters who want to exploit up-to-date streaming technology and will become a valuable reference to the specialized terminology, acronyms, and jargon associated with streaming. Many topics are treated with overviews highlighting the most important aspects of the topic and are accompanied by references to standards documents that provide the details needed by developers and other specialists. While much of this document will be useful to non-specialists, a thorough understanding of computer file structures and text encoding is required to comprehend some of the detailed examples provided herein.

When NRSC-G304 was adopted, Steve Shultis, New York Public Radio chaired the Data Services and Metadata (DSM) Subcommittee, and David Bialik, David K. Bialik & Associates, chaired the Metadata and Streaming Working Group (MSWG).

The NRSC-G304 drafting group consisted of the following members: David Bialik, Jeff Detweiler, John Kean, Frank Klekner, Scott Norcross, Greg Ogonowski, Robert Orban, and Steve Shultis. Also contributing to this document were Donna Detweiler, John Passmore, Dean Mitchell, Sam Sousa, and Conrad Trautmann, and others. The Working Group chair recognizes the extra efforts of Messrs. Ogonowski and Orban who developed the major portion of the text.

CONTENTS

FIGURES

TABLES

METADATA FOR STREAMING AUDIO HANDBOOK

1 SCOPE

This is an informative Handbook that provides guidance for broadcast engineers on audio streaming and metadata, focusing on how to use metadata with streaming audio services but also including over-the-air applications. Additionally, use of metadata with media files is discussed, the distinction being that media files contain audio (and possibly video) content that is downloaded for direct consumption on a device and is not streamed in realtime.

As broadcasters shift their focus to content creation for delivery across multiple distribution platforms, many have developed protocols to include video along with every audio segment they produce. This document focuses only on the audio parameters, as video standards have their own unique requirements.

A "Getting Started" guide is found in Section 4. Note that some of the technology described in this document is covered by patents and other intellectual property rights and may require licensing for use.

2 REFERENCES

2.1 Normative References

This is an informative Guideline. There are no normative references.

2.2 Informative References

The following references contain information that may be useful to those implementing this Guideline document. At the time of publication the edition(s) or version(s) indicated were valid. All references are subject to revision, and parties to agreements based on these references are encouraged to investigate the possibility of applying the most recent editions of the references listed below.

Additional references can be found in the text adjacent to pertinent topics.

[1] NRSC-G302, *Harmonization of Radio Metadata Across Transports Guideline*, National Radio Systems Committee, www.nrscstandards.org, January 2020.

[2] NRSC-G300-C, *RDS Usage Guideline*, www.nrscstandards.org, April 2018.

[3] EBU R-128, *Loudness normalisation and permitted maximum level of audio signals*, https://tech.ebu.ch/docs/r/r128.pdf.

[4] AES TD1008, *Recommendations for Loudness of Internet Audio Streaming and On-Demand Distribution,* https://www.aes.org/technical/documents/.

[5] IDE v2.4 specification

[6] NRSC-5-E, *IBOC Digital Radio Broadcasting Standard*, https://www.nrscstandards.org/.

[7] SCTE-35, *Digital Program Insertion Cueing Message*, https://www.scte.org/standards/library/catalog/scte-35-digital-program-insertion-cueing-message/.

[8] AES77, *Loudness Guidelines for Internet Audio Streaming and On-Demand Distribution*, https://www.aes.org/publications/standards/search.cfm?docID=114.

[9] ANSI/CTA-2075, *Loudness Standard for Over the Top Television and Online Video Distribution for Mobile and Fixed Devices*, https://www.cta.tech/Standards.

[10] ITU-R BS 1770-4

[11] R. Orban and G. Ogonowski, *Maintaining Audio Quality in the Broadcast/Netcast Facility*, www.indexcom.com/tech/audioquality/

[12] AES Audio Loudness Website: https://aes2.org/audio-topics/loudness-2/

[13] ISO/IEC 11172-3:1993: Information technology — Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s — Part 3: Audio https://www.iso.org/standard/22412.html

[14] ISO/IEC 13818-3:1998: Information technology — Generic coding of moving pictures and associated audio information — Part 3: Audio https://www.iso.org/standard/26797.html

2.3 Symbols and Abbreviations

In this Guideline the following abbreviations are used.

AAC Advanced audio coding
ADTS Audio data transport stream
AES Audio Engineering Society
AIF Audio Interchange File
AIFF Audio Interchange File Format
ALAC Apple Lossless Audio Codec
AM Amplitude modulation
ASCII American Standard Code for Information Interchange
ASP Active Server Pages
AVB Audio Video Bridging
CDN Content Delivery Network
CMAF Common media application format
CTA Consumer Technology Association
DASH Dynamic Active Streaming
DAI Direct Ad Insertion
EME Encrypted media extensions
ES Elementary stream
eSBR Enhanced Spectral Band Replication
FCC Federal Communications Commission (U.S.)
FLAC Free Lossless Audio Codec
FLV Flash video
FM Frequency modulation
fMP4 Fragmented MP4
GPS Global Positioning System
HLS HTTP live streaming
HDx HD Radio multicast channel designation (x=channel number, for example HD-1, HD-2, etc.)
HTML Hypertext markup language
HTTP Hypertext transfer protocol
HTTPS Secure Hypertext transfer Protocol
IBOC In-band/on-channel
ICY Icecast or SHOUTcast v1
IEC International Electrotechnical Commission
IETF Internet Engineering Task Force
ISO International organization for standardization
ISOBMFF **I**SO base media file format
LAN Local Area Network
MP3 MPEG-1 Audio Layer III or MPEG-2 Audio Layer III
MPEG Moving Picture Experts Group
MPS Main program service
MSE Media source extensions
MSWG Metadata & Streaming Workgroup
MPX Multiplex
MP4 MPEG-4 part 14
NAB National Association of Broadcasters
NRSC National Radio Systems Committee
OCA Open Control Architecture
OGG A container format maintained by the Xiph.Org Foundation
OTA Over-the-air
OTT Over-the-top

PAD	Program Associated Data
PHP	Hypertext Preprocessor
QR	Quick Response code
RBDS	Radio Broadcast Data Service
RDS	Radio Data System
RF	Radio Frequency
RFC	Request for comment
RIFF	Resource Interchange File Format
RTMP	Real-Time Messaging Protocol
RTSP	Real Time Streaming Protocol
SBR	Spectral Band Replication
SCTE	Society of Cable Telecommunications Engineers
SIS	Station information service
SQL	Structured Query Language
SRT	Secure Reliable Transport
SSAI	Single Server Ad Insertion
SSL	Secure Socket Layer
STL	Studio-to-transmitter link
TBD	To Be Determined
TLS	Transport Layer Security
TS	Transport Stream
URL	Uniform Resource Locator
UTF	Unicode Transformation Format
UTF-8	Universal Coded Character Set Transformation Format 8
USAC	Unified Speech and Audio Coding
VHF	Very High Frequency
VLAN	Virtual LAN
XML	Extensible Markup Language
WAN	Wide Area Network
WAV	Waveform Audio File Format (also **WAVE**)
WMA	Windows Media Audio

3 STREAMING MEDIA AND METADATA FUNDAMENTALS

"Streaming media" describes multimedia (audio and/or video) content sent continuously to internet-connected devices for real-time consumption. Streaming media exists primarily in two delivery categories: audio-only and video-with-audio. The method of delivery distinguishes "streaming" media from other types, which are delivered by sending the client a complete media file before rendering or playback can begin. Examples of downloaded media include delivery of HTML code and images to a web browser.

With streaming media, the client connects to the server or source, accumulates the incoming data in a buffer to accommodate variations in data flow, and renders the received content as it comes out of the buffer, usually without regard for when or if the stream will end. In this Guideline, this method is referred to as delivery in "realtime."[1] The content is generally not saved or stored after initial playback.

Instead of delivering a steady stream of small packets, modern streaming protocols like HLS take advantage of high-bandwidth connections to deliver larger packages of content in bursts. In HTTP live streaming (HLS and MPEG-DASH), the client downloads small, segmented files (often called "chunks") instead of receiving a constant flow of data. A manifest file tells the client how to assemble the chunks for continuous playout and allows the server to adapt the bit rate to changes in transmission bandwidth.[2]

Metadata content (text- or image-based) that describes the audio and/or video being streamed or downloaded is an important part of media delivery and can greatly enhance the user experience. Radio broadcasters are encouraged to provide rich metadata along with their audio content for both over-the-air and internet-based (streaming or downloading) delivery. For audio streaming, metadata can originate from a database or a separate file associated with the audio. The latter is more difficult to synchronize.

3.1 Streaming protocols and metadata delivery

In the early days of internet audio streaming, SHOUTcast[3] and Icecast[4] (ICY Protocol—"I Can Yell"), and the proprietary Real-Time Messaging Protocol in Flash Video (RTMP/FLV, now deprecated with Adobe's discontinuation of Flash) were dominant protocols. With these services, to prevent unwanted dropouts or even disconnections required a continuous, reliable connection from the encoder to the streaming server and from streaming server to the player client, which the internet was not designed to provide. The SHOUTcast/Icecast ICY streaming protocols were never standardized and there are many incompatible variants in use, requiring custom implementations.

SHOUTcast and Icecast use asynchronous, non-extensible metadata that cannot be precisely synchronized to the audio. The need for more extensible and robust metadata led to the use of out-of-band metadata, which may be delivered in parallel with the media stream or as an entirely separate metadata stream.

More recent streaming protocols such as HLS and MPEG-DASH address these limitations. Fully compliant HLS and MPEG-DASH implementations provide accurate metadata embedded in the stream. Because HLS leverages ID3V2.4 (a metadata protocol), the metadata is fully extensible, synchronous, on-time, and uses any UTF8 character set, including emoticons/emoji. Section 5 provides more detail on the features and advantages of HLS and MPEG-DASH.

A playout automation system typically supplies metadata to both the OTA transmission (Figure 1) and the streaming encoder (Figure 2). This metadata can be part of a data string that is then converted to a format consistent with the broadcast or streaming protocol. Several middleware products exist for conversion. An

[1] The playback of the streamed content is said to be mesochronous with the transmission of the stream because there is an unknown delay between the transmission from the server and the final buffering, rendering and playback at the client. The client must play the stream at the same rate as the source transmission.
[2] HTTP live streaming protocols are also mesochronous because the delivery of larger chunks of content must allow the client to render and play the content continuously. The chunking process adds a protocol layer that can increase latency while it provides additional degrees of freedom for the management of the stream to the client.
[3] https://www.SHOUTcast.com/
[4] https://icecast.org/

example of metadata is Now Playing Information, which can provide data for the listener as well as playout information for various reports.

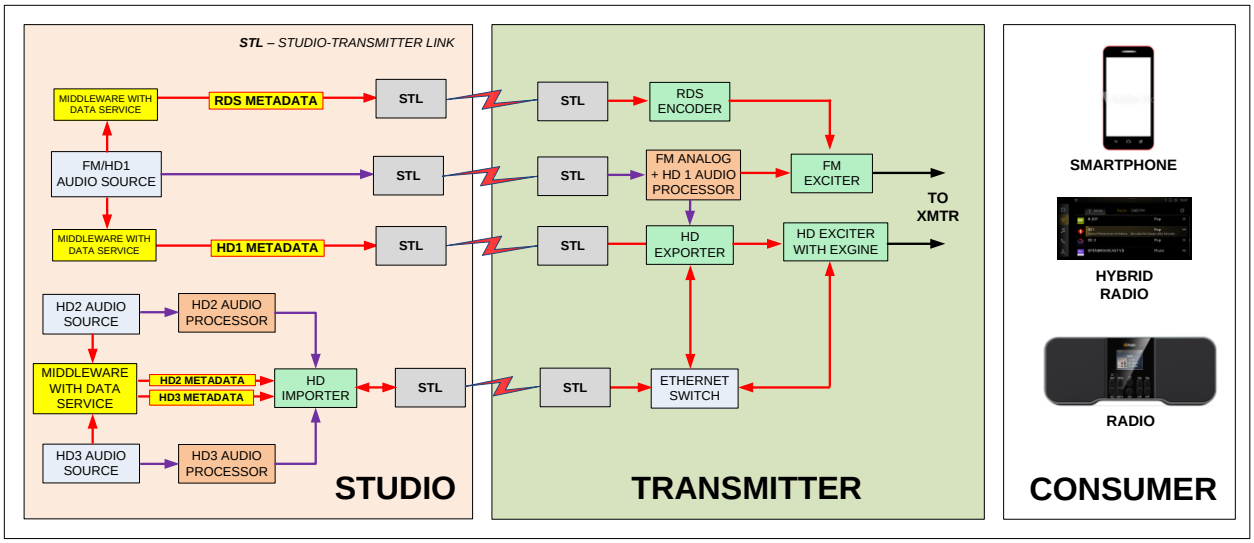


Figure 1. Diagram showing audio and metadata flow for OTA transmission.

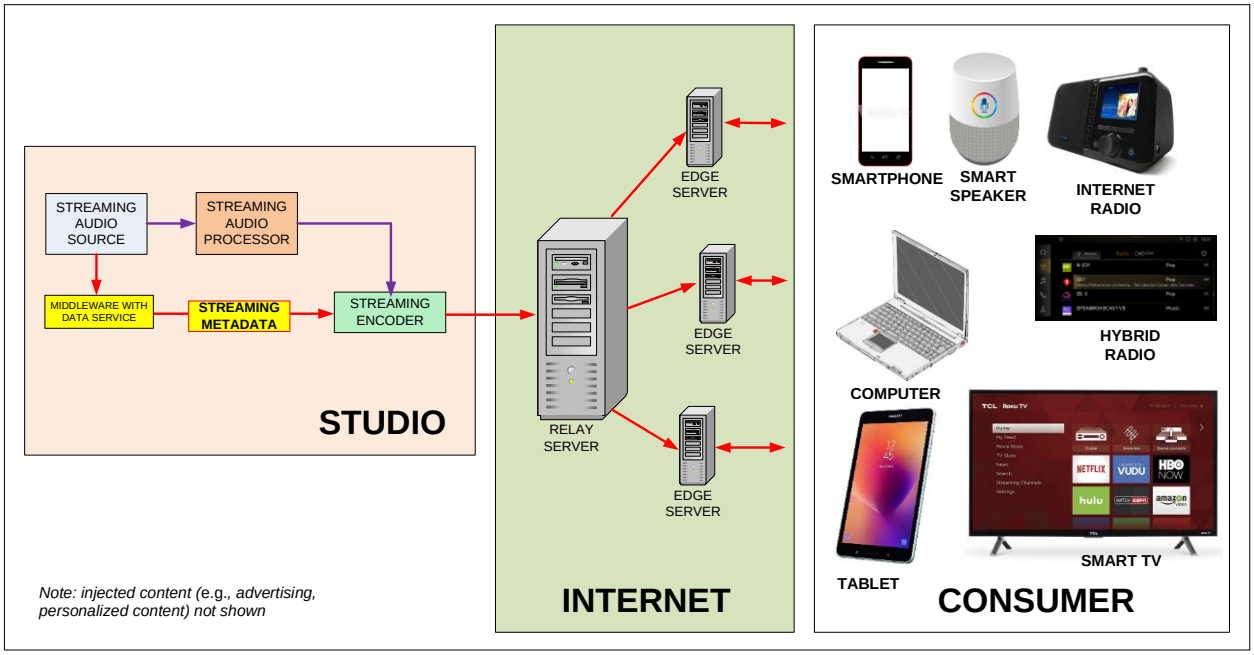


Figure 2. Diagram showing audio and metadata flow for internet streaming.

3.2 Metadata basics

Metadata is data that gives information about other data. Player devices may or may not recognize a given piece of metadata.

Metadata can describe structural information about the primary data in a stream, such as coding schemes, settings and available transmission rates. This is usually established in the transport protocol and is not the subject of this document.

In the content protocol, metadata also can be used to provide information related to the content of the media. For audio services, content-related metadata can specify text (identifying, for example, station name, song title and artist, program topic and host, etc.) or graphics (e.g., album art, station logo, advertisement image). Metadata can be static (such as the station name and logo) or dynamic (such as identifying the name of a program, advertiser, host, song, or performer).

Depending on the application, metadata may be used to provide rapidly updated content about or in parallel with the audio stream (listener reactions, tweets, or even music lyrics).

Operational metadata (such as loudness and dynamic range data) can enhance the user experience. Alternatively, streamers can use it internally for functions such as contact closures or ad insertion cues for the streaming server.

Table 1 lists some common metadata elements used by radio broadcasters.

Table 1. Commonly used metadata elements in radio broadcasting

Name	Type	Description	Origination
Stream Name	Text	Name of Stream/Station	Content provider
Stream Description	Text	Stream/Station description or sig	Content provider
Song Title	Text	Identifies song being played	Content provider
Song Artist	Text	Identifies artist	Content provider
Song Composer	Text	Identifies Composer	Content provider
Song Publisher	Text	Identifies Publisher	Content provider
Stream/Station Logo URL	URL	Image for Stream/Station Logo	Content provider
Album/Artist Image URL	URL	Image of artist or image of album art	Service provider (internet)
Commercial Image URL	URL	Image for commercials	Content/Service provider
Content Year	Number	Year of Content Release	Content provider
Loudness Values	Number	Optionally set audio level in player	Content provider
Dynamic Range Control	Number	Optionally set dynamic range control	Content provider
Content/Com Insert Control	Text	Cue point for network content insert	Content provider
Ext Remote Control	Text	External software/hardware control	Content provider

Metadata can support interactivity in web pages, where clickable links are conveyed to the web page via metadata. Section 14.1.4 provides details and examples.

3.3 Using a metadata service provider

Typically, program-associated metadata for radio programs is generated in a content provider's playout system. However, many broadcasters make use of metadata service providers to "condition" and augment the playout system-based metadata prior to broadcast. Metadata service providers can format the metadata for specific delivery mechanisms (e.g., RDS, HD Radio or internet-based audio streaming) and deliver these ready-to-go data streams to the broadcast, greatly simplifying the process for the broadcaster. Other advantages include:

- The playout system metadata may include notes or other information not meant for delivery to listeners, and metadata service providers can detect and remove this extraneous information prior to broadcast;
- There may be licensing costs associated with use of album art images, and metadata service providers can manage these licensing issues more efficiently that can individual broadcasters.

Potential disadvantages include:

- Increased latency due to communications overhead with the metadata service provider;
- Interfacing a playout system to a metadata service provider requires a custom API to convey title and artist from the playout system to the provider, which de-standardizes the playout and can add complexity and cost;
- For cuts that have been reissued multiple times with different album art, an image that is fetched based on title and artist alone may not correspond to the image associated with the specific cut in the playout system. Maintaining control of this process is an argument in favor keeping this process in-house. This is normally done using CD ripping software that offers a lookup server for album art. Details of licensing these images for broadcast or streaming are beyond the scope of this document.

3.4 Metadata capabilities of OTA radio broadcast services

The metadata capabilities of AM and FM radio services are dramatically different as illustrated in Table 2.

Table 2. Metadata capabilities of OTA AM and FM radio services

Service	Text-based metadata	Image-based metadata
Analog AM radio	Not supported	Not supported
Hybrid digital AM radio (HD Radio)	Supported	Not supported
All-digital AM radio (HD Radio)	Supported	Supported
Analog FM radio	Supported	Not supported
Hybrid digital FM radio (HD Radio)	Supported	Supported
All-digital FM radio (HD Radio)	Supported	Supported

NOTE 1: "hybrid digital" services utilize both analog and digital radio signals
NOTE 2: All-digital FM radio services are not currently authorized for use by the FCC
NOTE 3: RDS2, standardized by the IEC in 2019, will support image-based metadata for analog FM but is not currently in use in the US.

In analog FM services, the Radio Data System (RDS) digital FM subcarrier can provide textual metadata. RDS is a well-established metadata transport for analog FM broadcast. It is conveyed on a multiplexed 57 kHz subcarrier. Although the protocol originated in Europe in the 1980s, the NRSC first standardized it in 1993 as the Radio Broadcast Data Service (RBDS) and subsequently revised it several times. In 2021 the RDS and RBDS standards were harmonized into a single standard, with the RBDS portion being incorporated into an International Electrotechnical Commission (IEC) standard, IEC-62106-9.

RDS is useful for delivering text metadata to mobile listeners because most of today's automotive radios support RDS. FM broadcasters most typically use the Program Service (PS) and RadioText (RT) features to deliver messages to receiver displays. As originally envisioned, the PS field was to be displayed as a static 8-character string showing the station's call sign, but many broadcasters use the PS field dynamically to identify the station creatively, e.g., with slogans and other information. The RadioText field was designed for transmitting longer (64-character) text strings for artist and song title or other "now playing" metadata. RadioText+ is an enhancement to RadioText which allows for identification of text into specific fields, for example, song title or artist, instead of just being a nondescript string of characters.

Various aspects of RDS technology have been documented by the NRSC in standards and guideline documents:

- NRSC-G300-C, Radio Data System (RDS) Usage Guideline
- NRSC-G302, Harmonization of Radio Metadata Across Transports Guideline
- NRSC- G303, Best Practices for Delivering Emergency Alerts and Information for FM Radio Broadcasters

- NRSC-4-B United States RBDS Standard—Specification of the Radio Broadcast Data System, National Radio Systems Committee, April 2011 (now superseded by IEC 62106).
- IEC 62106, Radio Data System (RDS) - VHF/FM sound broadcasting in the frequency range from 64,0 MHz to 108,0 MHz - Part 9: RBDS - RDS variant used in North America.

HD Radio in-band/on-channel (IBOC) digital broadcasting technology provides a much greater metadata capability to broadcast radio services. Three specific services in particular - Station Information Service (SIS), Main Program Service (MPS) and Supplemental Program Service (SPS) - support metadata transmission pertaining to station information, main program audio and supplemental program audio (i.e., multicast channels), respectively. See [6] for additional information on the specific metadata supported by HD Radio. See section 10 for details on HD Radio metadata.

3.5 Metadata capabilities of streaming audio

As with an OTA signal, the source for metadata for the streaming audio version of a radio broadcaster's signal is the playout system. As is evident from Table 3, no matter which streaming format is used both textual and image metadata is supported.

Table 3. Metadata capabilities supported by some streaming formats used by radio broadcasters

Streaming format	Text-based metadata	Image-based metadata	Synchronous on-time delivery
HLS	Yes	Yes	Yes
MPEG-DASH	Yes	Yes	Yes
SHOUTcast	Yes	Yes	No
SHOUTcast 2	Yes	Yes	Yes
Icecast 2	Yes	Yes	No

3.6 Metadata for supplemental control and information

While metadata is most often used to provide information to clients about the media being delivered, it also provides broadcasters with a useful channel that allows integration with outside resources. Outside of providing stream information, the most common use of metadata at this writing is integration with advertising components. This metadata signals the streaming server when to insert advertisements into the media stream. Synchronous metadata can also be used as a command-and-control service, signaling servers and/or other hardware to take specific actions when specific metadata is received, such as starting and stopping recordings of the stream.

Using this kind of metadata presents challenges, such as accurate cue point alignment. It is imperative to synchronize metadata and audio for proper presentation. A synchronous streaming protocol such as HLS is necessary to achieve this by using its supported synchronous metadata; be sure not to use asynchronous metadata.

Successfully using metadata for this purpose requires precise time synchronization of all hardware involved. See section 8.

3.7 Loudness and dynamic range control metadata

While ID3 metadata is known and widely used for supplying information about the content associated with an audio file, another form of metadata identifies key characteristics of the digital audio itself. This audio-related metadata provides continuous information about the average loudness and dynamic range of the audio, and sometimes includes instructions that playback decoders can use to customize the loudness and playback dynamics as desired by each listener.

Audio-related metadata is used for the audio that accompanies most current streamed and broadcasted video content, offering listeners a choice of compelling, wide dynamic range sound or audio customized for a given player device and conditions such as late-night and mobile listening. Unlike regular ID3 data that identifies a song title or artist, dynamic range control (DRC) metadata requires fast and frame-accurate instructions so gain adjustments are made smoothly. This is accomplished by low-level frame-by-frame metadata within the codec. One application of DRC metadata is allowing listeners in noisy environments to manage their playback dynamic range so they can hear all parts of the music and understand all the speech, while other listeners in quieter environments can choose to hear the content in its original dynamic form, as intended by the artists. Two AES publications provide additional background:

- AESTD1008.1.21-9/AES77, *Recommendations for Loudness of Internet Audio Streaming and On-Demand Distribution*, provides recommendations for establishing and implementing an effective distribution loudness for streaming and on-demand audio file playback. It is intended for use by distributors of internet audio streams and on-demand audio files. These recommendations help preserve the audio quality of content and ensure that a listener can switch between streams without uncomfortable changes in loudness.

- ANSI/CTA-2075, *Loudness Standard for Over the Top Television and Online Video Distribution for Mobile and Fixed Devices*, while not directly applicable to this audio-only document, it is educational, providing detailed explanations of how loudness and dynamic range control metadata are implemented in the sound-for-picture provider/player ecosystem.

Loudness and dynamic range control metadata is implemented automatically in some codecs. As an alternative to codec metadata, loudness metadata can also be sent in content metadata and parsed with JavaScript. While this metadata can be incorporated into user-defined ID3 fields, it requires a custom player implementation because no universal standards exist at this writing. The closest such standard available is ReplayGain 2.0, which uses the ITU-R BS 1770-4 algorithm for loudness measurement.

File editors typically do not convey loudness metadata from their input files to their output files. The loudness metadata in the input file may be incorrect, and editing a file can change its Integrated Loudness, particularly if significant insertions or deletions occur. Therefore, loudness metadata should be regenerated after a file is edited. Several editing applications have built-in loudness analysis modules, which are convenient for this purpose.

While Integrated Loudness metadata can be added to a file manually, DRC metadata cannot be edited in an audio editor and must be generated automatically, usually by the codec when a file is encoded. Software that generates DRC metadata typically requires the user to specify a profile (or preset) that determines the nature and amount of dynamic compression that the player will produce.

A file's metadata must never contain loudness values different from the file's actual measured loudness. This can cause annoying file-to-file loudness variations when player devices normalize output level via metadata. Incorrect loudness metadata will also cause the DRC metadata to be generated incorrectly, possibly introducing undesirable artifacts if the consumer activates DRC-driven dynamic compression.

4 IMPLEMENTING METADATA IN A FACILITY—GETTING STARTED

This section provides an overview of how to implement metadata in a facility. In many facilities, metadata sources must feed both streaming and over-the-air (**OTA**) transmission, which have different levels of metadata support. Moreover, analog FM and HD Radio (see Metadata for HD Radio) have different metadata capabilities.

Streaming audio can be implemented in several different ways depending upon the source audio. Streams may originate from live studio operation or completely within a computer system, either local, network, or cloud. The concepts are similar except for a few different details and hardware configurations. In all cases, now-playing metadata is delivered to the streaming encoders in much the same way. Network protocols and appropriate security must be observed.

Refer to Figure 3 when reading the discussion that follows.

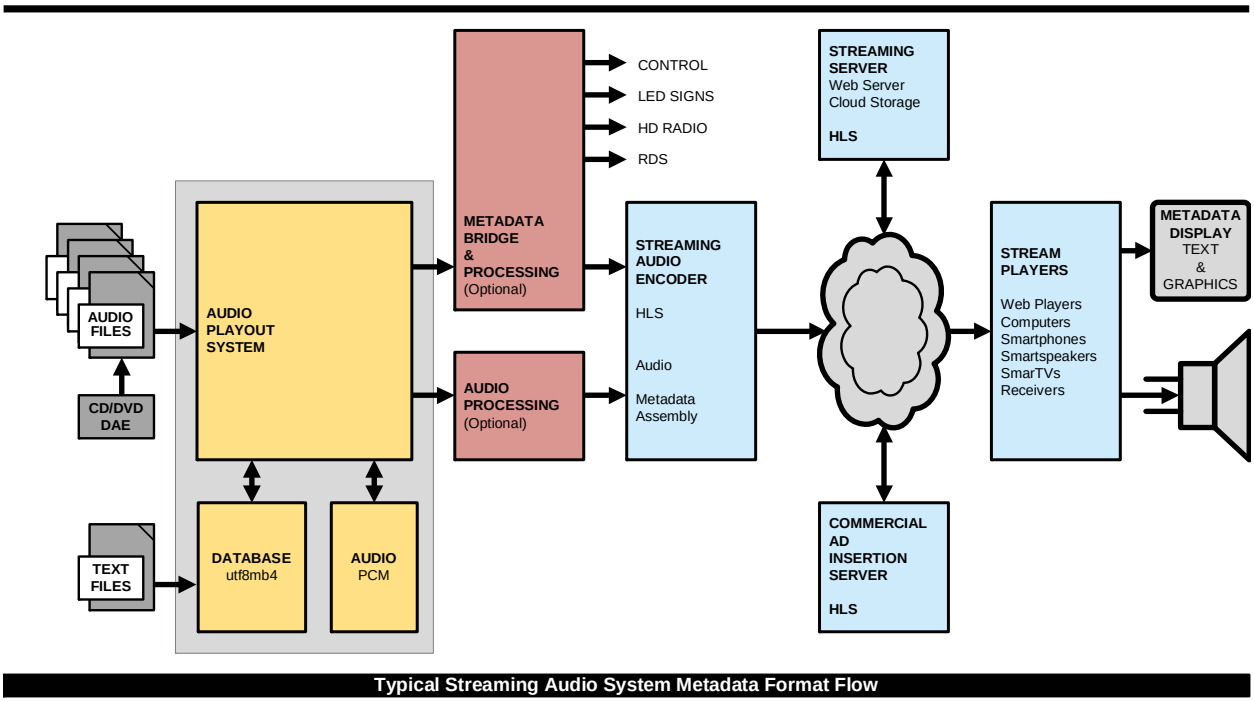


Figure 3. Typical simplified streaming audio system.

First, a decision must be made regarding how to deploy the stream. There are three basic approaches, listed in increasing order of complexity, to deploying HTTP(S) Live Streaming (HLS) and/or Dynamic Adaptive Streaming over HTTP (MPEG/DASH), the two protocols recommended in this document:

- Use a third-party, full-service Content Distribution Network (CDN). The CDN only requires audio and metadata, usually provided over the public internet. The CDN provides audience analytics, plus the formatting and bandwidth necessary to stream.
- Implement the streaming “front end” in-house but use a third-party web service provider like Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform to provide the analytics and also the scalable, raw data bandwidth needed to stream to your audience. In this case, the broadcaster operates the streaming formatter (HLS or MPEG/DASH) that multiplexes the audio and metadata. It sends the web service provider formatted data that is very similar to web pages.

- Implement streaming entirely in-house, operating streaming servers that convey content directly to audiences. This requires the most networking skills because the broadcaster must deploy the servers and generate analytics. Moreover, scaling server capacity to match audience demand may prove challenging.

Regardless of which approach chosen, HLS and/or MPEG/DASH can be readily deployed. When choosing a CDN, choose one that competently supports HLS and/or MPEG/DASH; some do not. Because this document is oriented toward metadata, it does not provide further detail on the general topic of setting up a stream.

Of course, implementing metadata transmission requires the metadata associated with a given piece of audio source content to be available to the playout system. Section 13 (Audio Media Files) discusses the metadata that different audio file formats and physical media can carry. If the source files ingested by the playout system are tagged with metadata, the playout system is responsible for parsing the metadata and updating the playout system's database automatically upon ingest. Section 13.8 (Audio Compact Disc File Extraction—CD Ripping) discusses how to obtain metadata when ripping CDs (which often contain no title/artist metadata) to playout systems. Note that different playout systems may have different methods and capabilities for managing their databases and parsing source metadata.

The playout system must be correctly configured to emit this metadata. Section 14 discusses in general terms how to do this and provides some detailed examples.

Optionally, Program Associated Data (PAD) bridge software receives the metadata from the playout system and formats it for various destinations, including streaming, analog over-the-air radio, and HD Radio. Alternatively, a third-party metadata service provider can be used. Sometimes it may be necessary to use one of these to transcode the Text Encodings depending on the encodings used in the source and destination, if the encodings used in the source and destination are not identical.

The audio and appropriately-formatted metadata (direct from the playout system, PAD bridge software, or a metadata service provider) are applied to the streaming audio encoder via an IP connection. The encoder multiplexes the metadata and audio and the result is conveyed to the audience as described above.

For over-the-air channels, appropriately-formatted metadata is conveyed via IP to a Radio Data System (RDS) encoder and optionally to the HD Radio transmission chain. (Some older-technology RDS encoders receive metadata via a serial connection.) The components in the HD Radio signal path and their configuration are application-specific and are beyond the scope of this document.

Other sections later in this document provide greater detail. Section 18 (Glossary and compendium of useful audio streaming terms) is a mini-encyclopedia providing reference material and definitions of various terms used in streaming and metadata technology.

Figure 10(Typical Streaming Audio System Metadata Flow) is an elaboration of Figure 3 (Typical Simplified Streaming Audio System) that shows the recommended data formats used to convey metadata through the entire broadcast and HLS or MPEG/DASH streaming signal paths.

Figure 4 shows examples of the rich graphic displays that can be achieved with creative use of the metadata capabilities available in HLS and MPEG/DASH. Using the UTF-8 character set allows correct display of non-Roman alphabets.

Figure 4. Examples of graphics achievable with HLS and MPEG/DASH metadata.

Figure 5 shows examples of issues that will occur if metadata is incorrectly implemented.

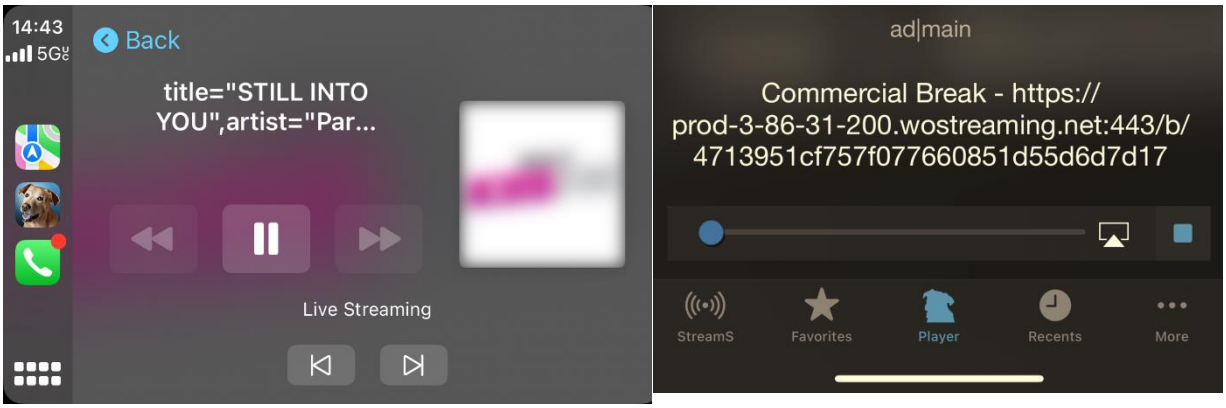


Figure 5. Examples of display errors caused by incorrect metadata implementation.

Competent HLS encoders allow using user-definable metadata for cueing and control without having to sacrifice the Artist and/or Title fields for insertion control. This prevents unwanted display of control metadata on consumer devices, as illustrated in the examples below. Additionally, the Title/Artist fields can then be exploited to create revenue by sending advertising images and logos to players during commercials.

4.1 Obtaining, Implementing and Maintaining SSL Certificates

Most broadcasters use a Content Delivery Network (CDN) for distributing their audio to the public. Like a station's Studio-to-Transmitter Link (STL) that connects their studio to a transmitter site, there must be a transmission method of getting the station's streaming media to the CDN.

Typically, the public internet is the most cost-effective solution used for this transmission path. SSL (Secure Socket Layer) certificates, or "Certs", are necessary to create a secure transmission path over the internet between a station's origin server and their CDN in a client/server topology. SSL is a type of encryption protocol that secures data between browsers and servers so it cannot be intercepted [see Secure Socket Layer (SSL)/Transport Layer Security (TLS) in the glossary].

Certificates contain a pair of keys: a public and a private key. Data encrypted with the public key can only be decrypted with the private key, and vice versa. Public-key encryption conceals the content of a message in a ciphertext that can only be decrypted with the private key. A handshake occurs between the public and private keys to consummate a secure connection.

A broadcaster must obtain the key pair from an official Certificate Authority (CA). It is important that a user obtain an SSL certificate is that it is digitally signed by a trusted CA. Anyone can create a certificate, but browsers only trust certificates that come from an organization on their list of trusted CAs.

SSL Certs do not last forever; when purchasing an SSL Cert, it is critical to note the term and expiration date of the certificate, as an expired certificate will no longer pass data. Some CDN's provide a warning service to their streaming clients to alert them that their cert is approaching its anniversary date so that the client can renew the cert. If they do not, the broadcaster must keep track of this themselves.

4.2 Player client implementations

Metadata is an integral part of player client implementation. While complete details are beyond the scope of this document, Annex A provides examples of communication between player clients and servers for HLS, MPEG-DASH, Icecast and SHOUTcast.

4.3 Handling metadata in production

Some applications allow the metadata to be edited/created. It is vital to preserve metadata needed for accounting/reporting and playback display, including now-playing information and graphics. Thus it is advisable to use an editor that accurately supports this functionality. As of this writing, many do not, or appear to support it but introduce errors. For example, the free open-source editor Audacity® preserves most FLAC metadata but loses images. The best dedicated external metadata applications[5], operated separately from the audio editor, are most likely to work correctly. Typically, these applications are available at modest cost.

Loudness metadata requires special handling. It is generally undesirable to copy loudness and DRC metadata from an input to an output file because modifying content may cause the loudness and DRC values in the input file to become invalid. This is discussed in section 3.7.

In addition to concerns about metadata, it is important to note that editing compressed files can cause transcoding to occur because the file must usually be decoded before it can be edited. With lossless encoding, this is nondestructive. However, if the file uses lossy encoding like AAC or MP3, transcoding will typically degrade audio quality.

If the application takes loudness metadata into account when decoding the input file, the user must consider the application's specific handling of this metadata—for example, does it apply loudness normalization when decoding the input file? For these reasons, files used to source program material for streaming and broadcast should be uncompressed, or, at worst, losslessly encoded, and the application's operator must fully understand a given organization's loudness management workflow. Unless one is creating a

dynamically compressed output file for an application like in-flight entertainment, it is undesirable for an editor to apply DRC when decoding an input file. This will apply irreversible dynamic compression to the audio data and thus defeat the whole purpose of DRC metadata, which is to preserve the original dynamic range of the program material and allow the final consumer to apply nondestructive compression in the player if desired.

4.4 QR codes directing to streams

QR codes used as scannable stream links allow the audience to reference, scan and connect directly to a stream without having to enter direct URLs. QR codes can be placed on web pages, billboards, stickers, or any other creative place.

The example QR code HTML image reference code below shows how to display a QR code on a web page and reference the audio stream.

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>QR Code Test</title>
</head>
<body>

<div id="qr1">
<a href="https://www.domain.com/hls/stream/program.m3u8" target="_new" >
<img onload="this.style.opacity='1'" src="img/qr1.svg"  />
</a>
</div>

</body>
</html>
```

QR Code HTML Stream Reference Code
This is for a direct connection to the stream from a dedicated player application.
This is an example of what is encoded in an HLS stream QR Code. It is for the StreamS HiFi Rdio App for iOS.

```
<hlsx://domain.com/hls/stream/program.m3u8>
```

A URI/URL scheme for the player application is registered to the host operating system.

4.5 Difference between HTML5 audio element, webaudio API and media source API

The functionality of an HTML5 Audio Element (<audio>) is significantly limited. For example, the Introduction paragraph of the W3C proposed standard for Web Audio states:

> The introduction of the audio element in HTML5 is very important, allowing for basic streaming audio playback. But it is not powerful enough to handle more complex audio applications.

This refers to the difference in capabilities between HTML5 Audio Elements and the more powerful WebAudio API and Media Source API:

- HTML5 Audio Element implements basic playback by embedding audio data in HTML. Although some browsers will play an ICY stream using this method, there is no metadata available, and it is subject to the unreliability of a constant stream. Many naïve developers choose this method because of simplicity. However, metadata must then be pushed out-of-band, making timing

unreliable. Moreover, some browsers fill their cache with audio, so long listening times cause problems with memory management, making the <audio> tag unsuitable for live ICY streams.

- WebAudio API is an ECMA Script (JavaScript) based API to support audio capabilities comparable to these of a modern stand-alone PC, such as mixing, processing and filtering tasks (e.g. reverb effects, distance attenuation, Doppler shift), related to audio production capabilities found in modern game audio engines, and streaming audio.
- The Media Source API, formally known as Media Source Extensions (MSE), provides functionality enabling plugin-free web-based streaming media. Using MSE, media streams can be created via JavaScript and played using <audio> and <video> elements, allowing precise media control and timing, including metadata.

5 METADATA FOR STREAMING AUDIO OVER HLS AND MPEG-DASH

Unlike legacy streaming protocols, HLS and MPEG-DASH use standards-based, extensible id3v2.4 metadata protocols. This allows a large amount of user-definable metadata to be sent. A given player client will only display the metadata it needs, and no standards-conforming player client will be broken by receiving metadata that it does not need. Furthermore, the recommended metadata method is in-band, within segments, so it is always on time and can be archived. It is thus useful for network-level content/ad insertion control and for on-time metadata display, including graphics, on consumer devices and dashboards. The metadata capabilities of HLS and MPEG-DASH exceed the capabilities of RDS or HD Radio.

5.1 Advantages of HLS and MPEG-DASH

HLS and MPEG-DASH streaming transport protocols are the NRSC-recommended approaches for streaming of broadcast radio content. One reason for this is that HLS and MPEG-DASH use segmented files for media file-based audio (on-demand audio and podcasts) and live streaming. They use the already available HTTP web delivery infrastructure. They do not require an internet connection to a dedicated streaming server, which is likely to fail eventually. This increases reliability and decreases operating cost.

When a player client is connected to a server delivering HLS, the network traffic appears identical to that of users clicking on web page elements within a web browser. Internet networks are highly optimized for this type of traffic, which HLS and MPEG-DASH methods leverage.

The server is typically either a low-cost, standard web server or cloud storage, both of which are highly reliable. The encoder ingests and assembles precise, on-time metadata and writes it to the streaming segments. Server infrastructures are easier and less costly to maintain because they are common web servers and/or cloud storage and do not require special streaming server expertise. High-performance HTML5 MSE player client resources are readily available, making HLS and MPEG-DASH implementations easier for less experienced web developers to deploy.

HLS and MPEG-DASH streaming encoders that support the SCTE-35 standard (which defines network-level and ad/content insertion) can control segment splitting and playlist generation, transferring this complex function from the content/ad providers.

Unlike some solutions that are designed to put the computational burden on the serving side of the link to relieve the client of the burden, HLS and MPEG-DASH are computationally intensive at both ends. Depending on the operating system and browser, browser-based players may use native implementations or HTML5 Media Source Extensions (MSE) and Encrypted Media Extensions (EME), relying on the power of the client device to handle the computational burden.

Software vetting is vital within the streaming infrastructure. HLS is a very wide, scalable protocol. Not all HLS and MPEG/DASH encoders support all of the features and advantages of these protocols so they must be chosen carefully.

Figure 6 is an illustration comparing metadata handling in HLS and ICY streaming methods. HLS and MPEG-DASH advantages over older protocols include:

- Higher resilience and reliability
- Considerably fewer points that contribute to latency in the audio and metadata paths from source to player
- Lower deployment cost
- Lower operating cost
- Precise on-time ad insertion
- Easily scales for large audiences.
- Because of its segmented nature, no player memory management problems regardless of how long the player has been playing a stream: Players download segment files, play them, and then delete them.
- Can run parallel encoders, making redundant encoder switching completely seamless and glitchless. (This is impossible with ICY.)

- Higher security
- Authentication and encryption content protection
- Higher audio quality due to seamless support of modern codecs and fewer audio dropouts
- Standards-based—RFC and ISO/CMAF—to maximize player compatibility
- Players start instantly
- Buffering dramatically minimized
- Uses standard network ports
- Adaptive bitrate
- High-performance synchronous and extensible metadata
    - Precisely-timed content/ad insertion metadata
    - Precisely-timed control metadata
    - Audio loudness and DRC metadata
- Uses a web server for live and file streams
- Uses inexpensive cloud storage for live and file streams
- Special stream server not required
- Hosts all media content on a single server
- HTML5-MSE player support with modern audio codecs including lossless and surround
- Live and file (on-demand) use the same server
- Live and file (on-demand) use the same player

5.2 NRSC recommendations for using HLS for audio streaming metadata

HLS Audio has two successive iterations:

- Original HLS Packed Audio uses Elementary Stream (ES) Segmented audio data transport stream (ADTS) at frame boundaries. ID3v2.4 frames may be inserted at audio codec frame boundaries. Multiple frames per segment are allowed as necessary. ES metadata should be included inside the audio segments and is then considered in-band. Audio-only streams should not use Transport Stream (TS), as the overhead for TS is excessive for audio-only.

- Updated HLS, known as CMAF audio-only, uses ISOBMFF Fragmented MP4 (fMP4), which makes the segments compatible with MPEG-DASH. This iteration supports new audio codecs such as Lossless and MPEG-D USAC with MPEG-D DRC (xHE-AAC™). ES does not support new codecs. Metadata uses the same id3v2.4 frame structure as original HLS Packed Audio, except it is in an ISO MP4 emsg Box with a timestamp. Multiple frames and timestamps per segment are allowed as necessary. fMP4 metadata is not in-line as is ES; it is timed for better accuracy. Both ES and fMP4 metadata should be included inside the audio segments and are then considered in-band.

This has all been standardized:

- IETF RFC 8216, HTTP Live Streaming, https://datatracker.ietf.org/doc/html/rfc8216
- https://datatracker.ietf.org/doc/html/draft-pantos-hls-rfc8216bis
- ISO/IEC 23000-19:2018, Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media, https://www.iso.org/standard/71975.html
- Carriage of ID3 Timed Metadata in the Common Media Application Format (CMAF), https://aomediacodec.github.io/id3-emsg/
- ID3 tag version 2.4.0 - Main Structure, https://id3.org/id3v2.4.0-structure
- ID3 tag version 2.4.0 - Native Frames, https://id3.org/id3v2.4.0-frames

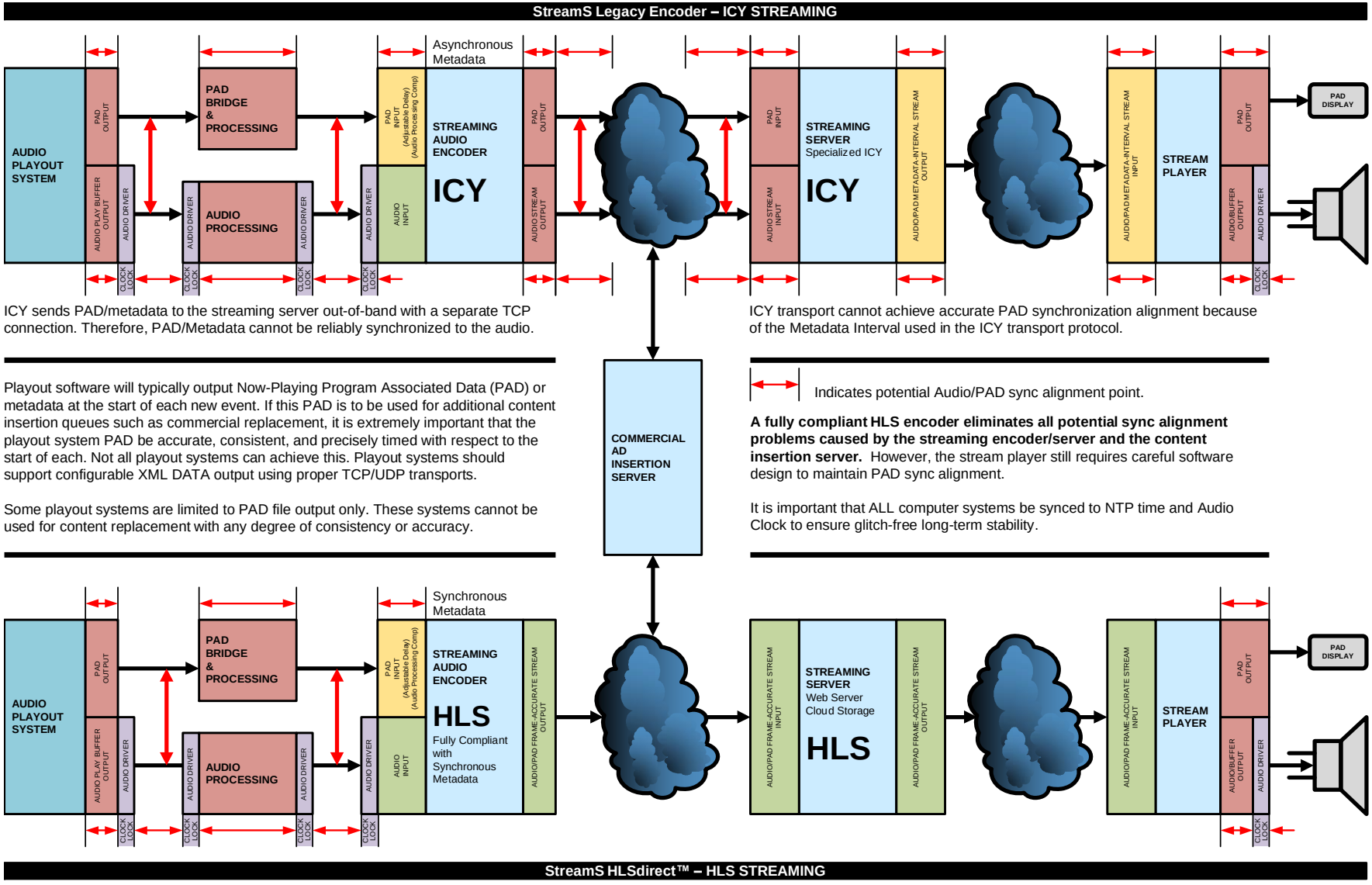


Figure 6. Diagram comparing metadata handling and associated latency in HLS and Icecast (ICY) streaming.

5.3 NRSC recommendations for using MPEG-DASH for audio streaming metadata

The current ISO CMAF MPEG-DASH metadata recommendations are the same as IETF HLS.

5.4 Diagrams comparing the three common methods of generating streams

The diagrams below show three common methods for streaming. Refer to the text in the diagrams for further details. Note that these can be zoomed as needed for legibility.

- Figure 7: HLS/DASH Encoder-generated file segment to HLS/DASH server-transferred file segment. This direct HLS is the preferred method, achieving all of the advantages of HLS/DASH shown in the list above. In this figure, TS stands for Time Stamp. Metadata in ES is placed in between audio frames. Metadata in fMP4 is placed with time stamps and is capable of more precise timing than ES.
- Figure 8: ICY-generated bitstream and metadata, transpacketized at the server to HLS/DASH. Despite its ultimate form as an HLS/DASH stream, this method starts with ICY, so it has many of the disadvantages of ICY, including asynchronous metadata and a fragile connection between encoder and server.
- Figure 9: ICY encoder-generated bitstream and metadata to an ICY streaming server. This is a common method in use, but its performance is the worst of the three, having none of the advantages of HLS/DASH. Its only real advantage is simplicity.

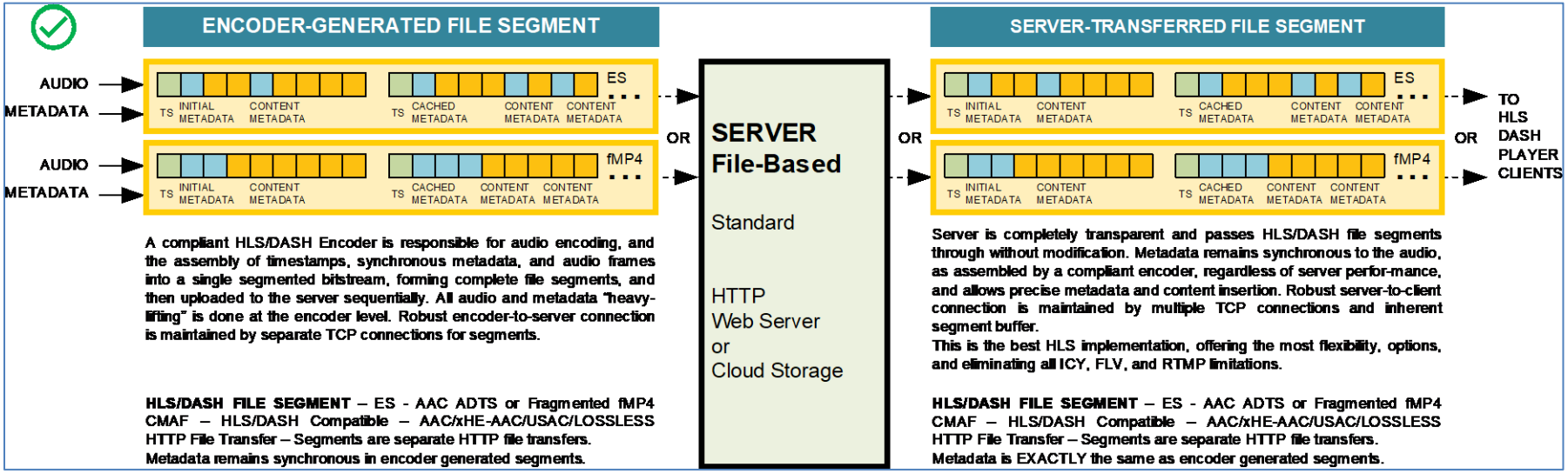


Figure 7. HLS/DASH Encoder-generated file segment to HLS/DASH server.

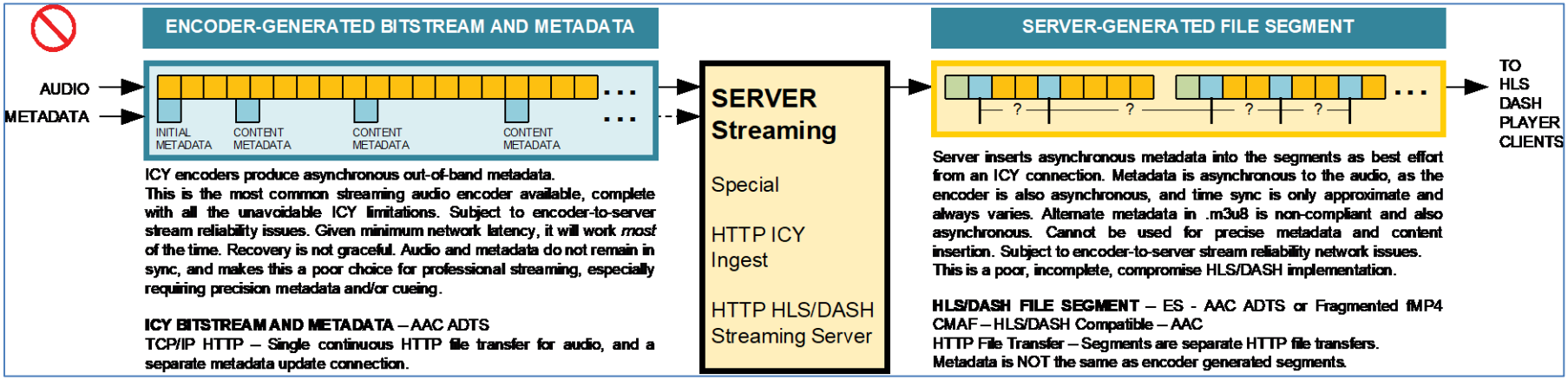


Figure 8. ICY-generated bitstream and metadata, transpacketized at the server to HLS/DASH.

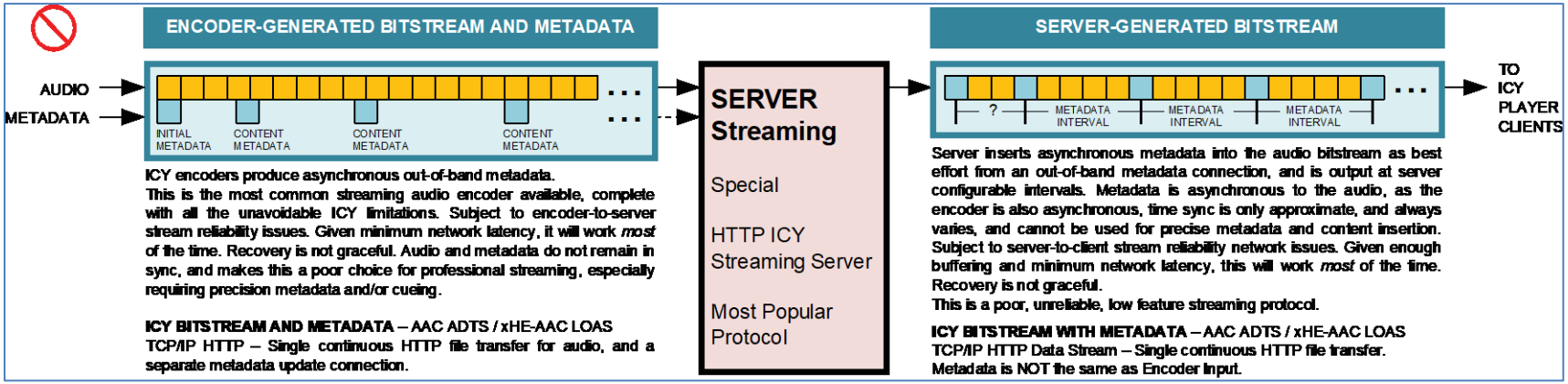


Figure 9. ICY encoder-generated bitstream and metadata to an ICY streaming server.

5.5 Data Formatting in Recommended Audio Metadata Flow

Figure 10 is an elaboration of Figure 3 that shows the recommended data formats used to convey metadata through the entire broadcast and HLS or MPEG/DASH streaming signal paths. These recommendations are standards-based and following them is likely to provide error-free results with players.

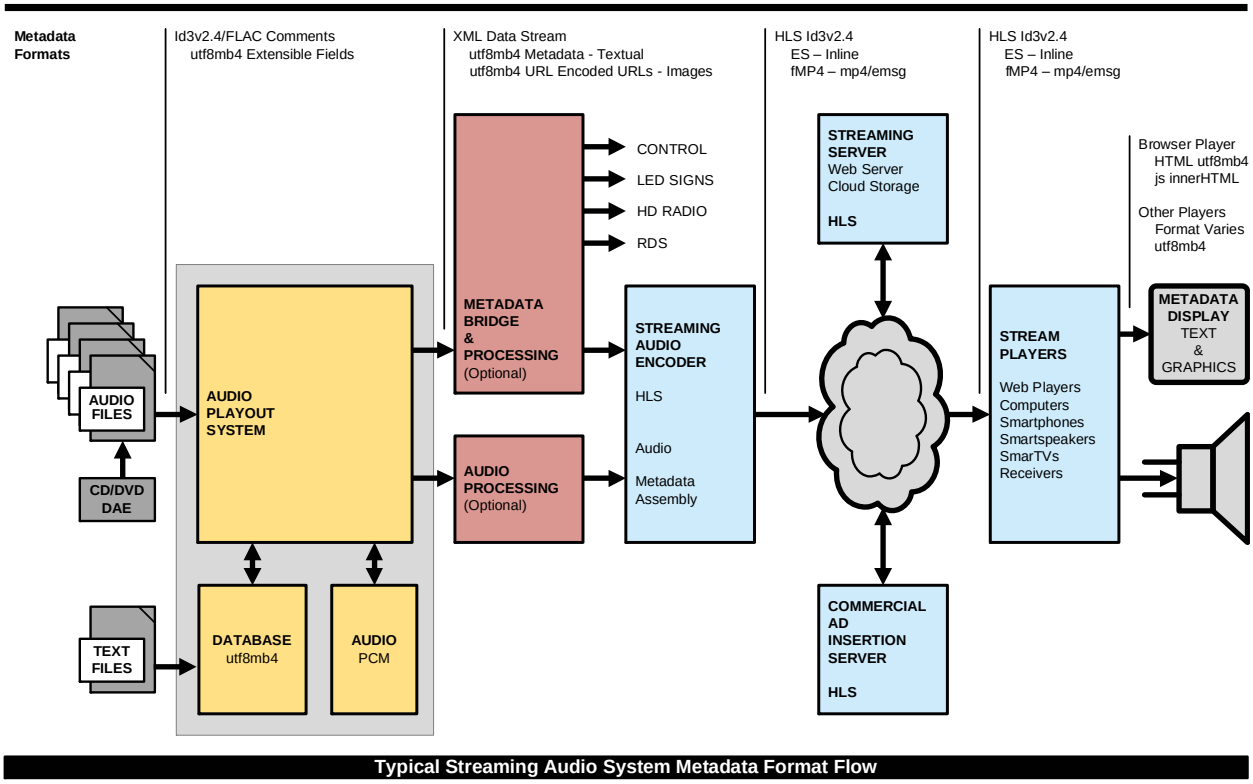


Figure 10. Typical streaming audio system metadata flow.
(Source: StreamS/Modulation Index/Greg Ogonowski)

The playout system files typically use ID3v2 metadata, or in the case of FLAC-encoded audio, UTF-8 Vorbis Comments.

The streaming audio encoder (and optional PAD bridge software) typically accepts from the playout system XML data that contain **UTF8mb4**-encoded text or images linked via UTF-8mb4-encoded URL Encoding.

The HLS encoder output uses ID3v2.4 embedded in an **Elementary Stream** (ES) or, an **emsg** Box in modern fragmented MP4 (**fMP4**).

The browser player internally parses the id3 data and turns it into **HTML** that is ready for display within the browser. Other player implementations will vary.

6 METADATA ENCODING

Metadata encoding requires several specialized techniques, discussed in the sections below.

6.1 PAD/Metadata Displayed on Web Pages using HTML

A possible application of PAD/Metadata is to display Now-Playing, Next Up, and/or Previously Played tracks on general-purpose web pages outside a player client. Such metadata is usually provided by a capable playout system and then needs to be sent to the web server. It must be HTML-encoded.

There are several ways to convey PAD/metadata to web pages. Security is very important because this requires transferring data from private networks over the public internet to a web server. Because of various security mechanisms implemented on web servers, without specialized PHP this will be a best-effort non-realtime display operation responding to a time-based query.

Sending Now-Playing info for display on a web page is usually a non-realtime process, as polling is usually the only method available. That said, this is typically accomplished by uploading a playout-system-produced XML file to the web server. The XML file is then parsed and polled by the web browser client using JavaScript.

To display Now-Playing Artist and Title on a web page, it is necessary to use HTML5 Encoding. Something as simple as the ampersand symbol (&) character is not sent as the “&” character. It is HTML-Encoded as &#38 (see Figure 11.

It is always preferable to declare the HTML document and encode the HTML document as **UTF8mb4** to handle special characters and character sets. The HTML reserved characters must *always* be HTML encoded.

There are several challenges with rendering Now-Playing metadata to a webpage correctly. These include security, permissions, and enabled webserver features such as PHP or ASP. Most third-party webservers do not allow any of this. Furthermore, it takes custom PHP or ASP and JavaScript to make all this work correctly, requiring senior programming expertise. Solutions should be available from any competent streaming encoder vendor; choose your vendor carefully. The PHP, ASP, and JavaScript for metadata are beyond the scope of this document.

```
<div>Hot &#38; Cold – Katy Perry</div>
```

Figure 11. HTML encoding example.

If the metadata workflow is done correctly, it will be difficult to see the HTML Encoding in the HTML source code, as the HTML Encoding is all in dynamic JavaScript. Although it is difficult to see in the source code, it will certainly display on the web page.

6.2 URL Encoding

When sending PAG: Program Associated Graphics with Now-Playing metadata, it is necessary to send a URL in metadata. Because a URL uses a limited character set and includes reserved characters that cannot be used, these URLs *must* then be URL-Encoded. This is not only true for International Character Sets, but for English as well. It is not necessary to encode all of the characters in the string, such as https://domain.com/.

For examples, see section 7.2 (URL Encoding).

6.3 Limitations of Title and Artist metadata derived from filenames

In naming filenames or folders, do not use any of the illegal characters or symbols shown in Figure 12. A properly designed and configured playout system should store characters in a Unicode database. This allows these characters to be used in metadata, as well as any Unicode character set including emoji/emoticons. Realtime metadata is then correctly transmitted.

#	pound
%	percent
&	ampersand
{	left curly bracket
}	right curly bracket
\	back slash
<	left angle bracket
>	right angle bracket
*	asterisk
?	question mark
/	forward slash
	blank spaces
$	dollar sign
!	exclamation point
'	single quotes
"	double quotes
:	colon
@	at sign
+	plus sign
`	backtick
|	pipe
=	equal sign

Figure 12. Illegal characters and symbols that should not be used in naming files or folders.

6.4 Examples of Titles and Artists that cannot be correctly encoded using filenames

Len Barry – 1 – 2 – 3
Which hyphen do you parse on? Maybe the first? What happens if the Artist has a “space hyphen space”?

Some content providers have decided to custom-format this field, which breaks standard players and can cause them to display garbage characters.

The following short list of Artist and Titles demonstrates the reason for using a database instead of filenames. Simple operating system filenames cannot hold this kind of information; databases can.

? and The Mysterians - 96 Tears
Love Potion #9 - Searchers
Ke$ha
Hall & Oates
Rinôçérôse
Axwell Λ Ingrosso - Something New
XYLØ
P!nk
MØ

54●40* - One Day In Your Life
Does Anyone Really Know What Time It Is? - Chicago
Why? - Bronski Beat
Electric Love - BØRNS
In the Room: Cruisin'
Gallant & Andra Day
歡聚歌 - New Formosa Band
w/o u - GEMS
Hælos - Dust
이게 무슨 일이야 - B1A4
Hush Hush; Hush Hush - The Pussycat Dolls
พริบตา (feat. Stamp) - เบิร์ด ธงไชย
Outlandish - Warrior//Worrier

6.5 Limitations of Icecast Metadata

The Icecast streaming protocol metadata has a limited feature set. For example, it contains one single field for both Artist and Title separated with a hyphen. This causes problems for several reasons.

Using a hyphen as a delimiter makes it impossible to accurately parse and separate Artist and Title for those applications requiring so. However, modern applications require much more than just displaying Artist – Title together in a simple player. Player devices and applications have evolved to provide much more capability.

7 TEXT ENCODING

7.1 When Text is Not Just Text

Metadata that uses characters beyond the basic ASCII set requires text encoding that is compatible with desired target applications and player devices. This is a complex subject, as several standards are in common use. Encoding errors cause metadata to be incorrectly displayed at the player device, often in obvious ways that compromise consumer perception of the stream's quality. It can also cause embedded URLs to fail.

Instead of characters, it is actually more correct to refer to "code points" when discussing text encoding systems. Code points allow abstraction from the term "character" and are the atomic unit of storage of information in an encoding. Most code points represent a single character, but some represent information such as formatting and control.

In this document, unless otherwise specified, "UTF-8" (8-bit Unicode Transformation Format) refers to UTF8mb4. UTF-8 is a modern standard that dates from 1992-1993. It is capable of encoding all 1,112,064 valid character code points in Unicode by using one to four one-byte (8-bit) code units. Its goal is to encompass every character in every language in the world: as of this writing, it supports more than 107,000 characters found in 90 writing systems.

Code points with lower numerical values, which tend to occur more frequently, are encoded using fewer bytes. UTF-8 was designed for backward compatibility with ASCII. The first 128 characters of Unicode, which correspond one-to-one with ASCII, are encoded using a single byte with the same binary value as ASCII so that val`id ASCII text is valid UTF-8-encoded Unicode as well. Because it is so comprehensive and efficient, UTF-8 is now widely recommended and steadily becoming the standard way to represent text in files, email, web pages, and software.

ISO-8859-1 was formerly the recommended character encoding for internet and web pages but is now deprecated and replaced by UTF-8. In addition to ISO-8859-1, there are many other ISO-8859 encodings to complicate matters. There are also many other types of encodings available, including Windows Code Pages, all of which are now unnecessary with HTML5 and UTF-8.

The examples in sections 7.1.1 through below show different encodings (identified by the subsection labels) of the same characters.[6] The examples start with "Test 123" (encodable with ASCII and thus compatible with all character encodings shown). They continue with characters that are outside the ASCII character set and are thus potentially troublesome. Characters in shown red cannot be encoded correctly for the specified character encoding. To fully understand the examples, the actual encoding must be examined in a hex editor as shown by the hex display following the example characters.

The examples were prepared in a UTF-8-compliant text editor that can transcode a text file using several different output encodings. The five lines of text in the example were first entered in the editor in UTF-8 encoding. The editor was then instructed to output the text file in the encodings shown in the examples below. It automatically identified characters that could not be represented in a given encoding and replaced them with question marks (hex 3F, shown in red in the examples). Note that in the examples below, the Microsoft Windows CR/LF character sequence is represented by hex `0D 0A`.

In summary, decoding text with the incorrect decoding will render incorrect characters. This is the reason why Now-Playing metadata is sometimes incorrectly displayed. As the limitations in the examples below illustrate, use UTF8mb4 encoding for everything, as it accommodates all possible characters in the most standard way available.

7.1.1 UTF-8

Compatible with the first 128 US-ASCII characters.

[6] The examples in the section are courtesy of Modulation Index LLC.

Test 123
ñ
€
™
😀

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  54 65 73 74 20 31 32 33 0D 0A C3 B1 0D 0A E2 82  Test 123..Ã±..â‚
00000010  AC 0D 0A E2 84 A2 0D 0A F0 9F 98 80              ¬..â„¢..ðŸ˜€
```

7.1.2 UTF-8 BOM

BOM stands for Byte Order Mark. UTF-8 BOM is compatible with the first 128 US-ASCII characters. It is distinguished from ordinary UTF-8 by three hidden characters at the beginning of the document to indicate that subsequent characters are encoded in UTF-8.

Test 123
ñ
€
™
😀

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  EF BB BF 54 65 73 74 20 31 32 33 0D 0A C3 B1 0D  ï»¿Test 123..Ã±.
00000010  0A E2 82 AC 0D 0A E2 84 A2 0D 0A F0 9F 98 80     .â‚¬..â„¢..ðŸ˜€
```

7.1.3 UTF-16BE

BE stands for Big-endian. UTF-16BE is not compatible with UTF-8. There are three hidden characters at the beginning of the document to indicate that subsequent characters are encoded in UTF-16BE.

Test 123
ñ
€
™
😀

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  FE FF 00 54 00 65 00 73 00 74 00 20 00 31 00 32  þÿ.T.e.s.t. .1.2
00000010  00 33 00 0D 00 0A 00 F1 00 0D 00 0A 20 AC 00 0D  .3.....ñ.... ¬..
00000020  00 0A 21 22 00 0D 00 0A D8 3D DE 00              ..!".....Ø=Þ.
```

7.1.4 UTF-16LE

LE stands for Little-endian. UTF-16LE is not compatible with UTF-8. There are three hidden characters at the beginning of the document to indicate that subsequent characters are encoded in UTF-16LE.

Test 123
ñ
€
™

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	FF	FE	54	00	65	00	73	00	74	00	20	00	31	00	32	00	ÿþT.e.s.t. .1.2.
00000010	33	00	0D	00	0A	00	F1	00	0D	00	0A	00	AC	20	0D	00	3.....ñ.....¬ ..
00000020	0A	00	22	21	0D	00	0A	00	3D	D8	00	DE					..\"!....=Ø.Þ

7.1.5 US-ASCII

128 characters representing the English alphabet and including some control and non-printable characters.

Test 123
?
?
?
??

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	54	65	73	74	20	31	32	33	0D	0A	3F	0D	0A	3F	0D	0A	Test 123..?..?..
00000010	3F	0D	0A	3F	3F												?..??

7.1.6 Windows-1252

Windows-1252 is a Microsoft proprietary character set. It is often informally called “ANSI,” although “ANSI” also has been used to denote Code Page 437, the character set of the original IBM PC. Windows-1252 is a superset of ISO-8859-1 with the addition of 27 characters in locations that ISO assigns to control codes. Apple’s proprietary MacRoman character set (no examples of which are provided in this document) contains a similar variety of characters from 128 to 255, but very few of them are assigned the same numbers as Windows-1252. MacRoman also assigns characters to the control-code positions.

The following example shows Windows-1252 encoding:

Test 123
ñ
€
™
??

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F

```
00000000  54 65 73 74 20 31 32 33 0D 0A F1 0D 0A 80 0D 0A  Test 123..ñ..€..
00000010  99 0D 0A 3F 3F                                    ™..??
```

7.1.7 ISO-8859-1 / Latin 1

All US-ASCII and additional characters.

Test 123
ñ
?
T
??

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  54 65 73 74 20 31 32 33 0D 0A F1 0D 0A 3F 0D 0A  Test 123..ñ..?..
00000010  54 0D 0A 3F 3F                                    T..??
```

7.2 URL Encoding

URL encoding is necessary to send image URLs in metadata sometimes within XML data, unless image filenames contain only ASCII characters. However, spaces are not allowed and must be encoded as: %20

Four examples are provided below:

https://domain.com/img/1/Rinôçérôse.jpg

```
https://domain.com/img/1/Rin%C3%B4%C3%A7%C3%A9r%C3%B4se.jpg
```

https://domain.com/img/1/04 Tees & Jeans.jpg

```
https://domain.com/img/1/04%20Tees%20&%20Jeans.jpg
```

https://domain.com/img/1/05 Machine Pour Les Oreilles.jpg

```
https://domain.com/img/1/05%20Machine%20Pour%20Les%20Oreilles.jpg
```

https://domain.com/img/1/山下達郎_クリスマス・イブ -English Version-_クリスマス・イブ (30th Anniversary Edition)_0-03.jpg

```
https://domain.com/img/1/%E5%B1%B1%E4%B8%8B%E9%81%94%E9%83%8E_%E3%82%AF%E3%83%AA%E3%82%B9%E3%83%9E%E3%82%B9%E3%83%BB%E3%82%A4%E3%83%96%20-English%20Version-_%E3%82%AF%E3%83%AA%E3%82%B9%E3%83%9E%E3%82%B9%E3%83%BB%E3%82%A4%E3%83%96%20(30th%20Anniversary%20Edition)_0-03.jpg
```

7.3 Example of an HTTP Query Playout Metadata Output

The example below shows what is typically sent to a web server for the display of Now-Playing information on a web page..This is generated from playout or bridge software. The HTTP Query method in the web page sends requests to the web server during the execution of the page opens.

URL Encoding is also required if HTTP query strings are used to send metadata to web servers. These are usually used with custom server-side script processing, such as PHP or ASP, to further author the HTML,

transforming the appropriate URL into HTML entries. Alternatively, it is common to use an internal or external image database to supply graphics.

```
https://domain.com/metadata.php?artist=Artist%20Goes%20Here&title=Title%20Goes%20Here&image=https://domain.com/img/Rin%C3%B4%C3%A7%C3%A9r%C3%B4se.jpg&next1_artist=Next%20Artist%20Goes%20Here&next1_title=Next%20Title%20Goes%20Here
```

7.4 Example of an HTTP Query Playout Metadata Output Template

Playout system: RadioDJ Pro.

The image URL, which points to the actual image, is URL-encoded using UTF-8.

```
artist=$artist_enc$&title=$title_enc$&image=https://domain.com/img/$image_enc$&next1_artist=$next1_artist_enc$&next1_title=$next1_title_enc$
```

Note that the domain is included in the template and is not specified as part of the image file variable. This expedites changing image file server locations, if necessary, by editing the template in one place.

7.5 HTML5 Encoding

To render all characters correctly, UTF-8 should be used for displaying Now Playing information in HTML5 documents and web pages. Because the document should be declared UTF-8, HTML encoding is unnecessary. The encoding declaration can also be configured at the web server. These details are beyond the scope of this document.

```
Content-Type: text/html; charset=UTF-8
```

HTML versions prior to HTML5 and/or alternate declarations are deprecated. If it is nevertheless necessary to use non-HTML5, note that doing so requires UTF-8 HTML encoding, either for encoding only unsafe (non-ASCII) characters or for encoding the entire string, which is less efficient. There are also many other types of encodings available, including Windows Code Pages, all of which are now unnecessary with HTML5 and UTF-8.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>HTML UTF-8 Encoding</title>
</head>
<body>
<p>
<strong>UTF-8</strong><br>
山下達郎 クリスマス・イブ -English Version- クリスマス・イブ (30th Anniversary Edition)
</p>
<p>
<strong>UTF-8 HTML Encoded - unsafe and non-ASCII characters only</strong><br>
&#x5C71;&#x4E0B;&#x9054;&#x90CE; &#x30AF;&#x30EA;&#x30B9;&#x30DE;&#x30B9;&#x30FB;&#x30A4;&#x30D6; -English Version- &#x30AF;&#x30EA;&#x30B9;&#x30DE;&#x30B9;&#x30FB;&#x30A4;&#x30D6; (30th Anniversary Edition)
</p>
<p>
<strong>UTF-8 HTML Encoded - all characters</strong><br>
&#x5C71;&#x4E0B;&#x9054;&#x90CE;&#x20;&#x30AF;&#x30EA;&#x30B9;&#x30DE;&#x30B9;&#x30FB;&#x30A4;&#x30D6;&#x20;&#x2D;&#x45;&#x6E;&#x67;&#x6C;&#x69;&#x73;&#x68;&#x20;&#x56;&#x65;&#x72;&#x73;&#x69;&#x6F;&#x6E;&#x2D;&#x20;&#x30AF;&#x30EA;&#x30B9;&#x30DE;&#x30B9;&#x30FB;&#x30A4;&#x30D6;&#x20;&#x28;&#x33;&#x30;&#x74;&#x68;&#x20;&#x41;&#x6E;&#x6E;&#x69;&#x76;&#x65;&#x72;&#x73;&#x61;&#x72;&#x79;&#x20;&#x45;&#x64;&#x69;&#x74;&#x69;&#x6F;&#x6E;&#x29;
</p>
</body>
</html>
```

The above text may be copied into a UTF-8 HTML file and tested within browsers.

7.6 HTML Error Pages

Note the ISO-8859-1 encoding declarations.

These are used to remain backward compatible with HTML versions prior to HTML5, supporting older browsers and devices.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head>
  <title>404 Not Found</title>
 </head>
 <body>
  <h1>404 Not Found</h1>
 </body>
</html>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>404 - File or directory not found.</title>
</head>
<body>
<div id="header"><h1>Server Error</h1></div>
<div id="content">
 <div class="content-container"><fieldset>
  <h2>404 - File or directory not found.</h2>
  <h3>The resource you are looking for might have been removed, had its name changed, or is temporarily
unavailable.</h3>
 </fieldset></div>
</div>
</body>
</html>
```

Provided by StreamS/Modulation Index, LLC

7.7 Example XML Playout Metadata Output for Data Stream

Playout system: RadioDJ Pro.

This is typically what is sent to a streaming audio encoder as realtime data (as opposed to a file) for Now-Playing information.

Note that the image URL is URL-encoded with UTF-8.

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[Rinôçérôse]]></artist>
        <title><![CDATA[Machine Pour Les Oreilles]]></title>
        <trackid>1516</trackid>
        <duration>00:05:40</duration>
        <album><![CDATA[✨This & That]]> 1999</album>
        <year>1999</year>
        <image>https://domain.com/img/1/05%20Machine%20Pour%20Les%20Oreilles.jpg</image>
        <accompaniment><![CDATA[NEXT: The Head and the Heart - Let's Be Still]]></accompaniment>
        <composer>Composer</composer>
```

```
        <group>Group</group>
        <isrc></isrc>
        <category>Music</category>
        <audio_gain>1.000</audio_gain>
        <ext>
                <item>
                        <key>ext_01</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_02</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_03</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_04</key>
                        <value>0</value>
                </item>
        </ext>
</Event>
```

Provided by StreamS/Modulation Index, LLC

7.8 Example XML Playout Metadata Output for Data Stream (Japanese and English)

Playout system: RadioDJ Pro
Japanese and English

This is typically what is sent to a streaming audio encoder as realtime data (as opposed to a file) for Now-Playing information.

The image URL is URL-encoded with UTF-8.

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[山下達郎]]></artist>
        <title><![CDATA[クリスマス・イブ -English Version-]]></title>
        <trackid>2312</trackid>
        <duration>00:04:04</duration>
        <album><![CDATA[✨This & That]]> 2013</album>
        <year>2013</year>
        <image>https://domain.com/img/1/%E5%B1%B1%E4%B8%8B%E9%81%94%E9%83%8E_%E3%82%AF%E3%83%AA%E3%82%B9%E3%83%9E%E3%82%B9%E3%83%BB%E3%82%A4%E3%83%96%20-English%20Version-_%E3%82%AF%E3%83%AA%E3%82%B9%E3%83%9E%E3%82%B9%E3%83%BB%E3%82%A4%E3%83%96%20(30th%20Anniversary%20Edition)_0-03.jpg</image>
        <accompaniment><![CDATA[NEXT: Space - Lost In Space]]></accompaniment>
        <composer>Composer</composer>
        <group>Group</group>
        <isrc>JPWP01371091</isrc>
        <category>Music</category>
        <audio_gain>1.000</audio_gain>
        <ext>
                <item>
                        <key>ext_01</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_02</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_03</key>
                        <value>0</value>
```

```
                </item>
                <item>
                        <key>ext_04</key>
                        <value>0</value>
                </item>
        </ext>
</Event>
```

7.9 Example XML Playout Metadata Output for Data Stream Template

Playout system: RadioDJ Pro.

The example below shows what is typically used to generate the XML data for Now-Playing information.

The image URL is URL-encoded using UTF-8.

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[$artist$]]></artist>
        <title><![CDATA[$title$]]></title>
        <trackid>$track_id$</trackid>
        <duration>$duration$</duration>
        <album><![CDATA[$album$]]> 2013</album>
        <year>$year$</year>
        <image>$image_enc$</image>
        <accompaniment><![CDATA[NEXT: $next1_artist$ - $next1_title$]]></accompaniment>
        <composer>$composer$</composer>
        <group>$subcat_name$</group>
        <isrc>$isrc$</isrc>
        <category>$track_type_name$</category>
        <audio_gain>1.000</audio_gain>
        <ext>
                <item>
                        <key>ext_01</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_02</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_03</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_04</key>
                        <value>0</value>
                </item>
        </ext>
</Event>
```

Note that the domain is included in the template and is not specified as part of the image file variable. This expedites changing image file server locations, if necessary, by editing the template in one place.

7.10 RDS Encoding

The RDS character set is not the same as any character set used in the computer world, such as ASCII, ISO-8859-1 (Latin 1), UTF-8 etc. The RDS character set has many characters in common with ISO-8859-1, especially in the first 128 positions (lower 7 bit characters). However, PAD bridge software or third-party PAD services must take the differences into account.

Playout system data will typically be encoded differently from RDS encoding. Obviously, these data must be translated to RDS character set data for transmission.

The most common character translation problem is that some radios display the wrong character for the dollar sign (“$”). Table 4 shows “$” the character maps for various standards. Other character sets place the “$” at 0x24, while RDS places the “$” at 0xAB.

Table 4. Character set translations for common dollar sign characters

Character Set	0x24	0xAB
RBDS/RDS (IEC 62106 Table E.1)	¤	$
ASCII / ISO 646 1994 / ECMA 6 7-bit Char Set	$	n/a
ISO-8859-1 Latin 1	$	«
Arial font set (or various other font sets)	$	«

The two most common faulty behaviors with “$” are:

1) Receivers sometimes display the ISO-8859-1 or Arial font equivalent graphic for 0xAB (which is ‘«’) instead of the RDS Standard graphic at 0xAB (the U.S. dollar sign, “$”). However, other character sets have also been observed on radios;

2) Broadcasters sometimes use the ISO-8859-1 value of 0x24 for transmission of ‘$’, leading to RDS-compliant radios displaying the international currency symbol, “¤”. Broadcasters should instead transmit 0xAB for the U.S. dollar sign ‘$’ in the RDS character map.

Note that the dollar sign is not the only character that is not the same in RDS as in encodings such as UTF-8. See section 9 of [2] for more information.

7.11 Endianness

Big-endian (BE) and little-endian (LE), are two different methods of organizing multi-byte binary data in computer memory. They refer to the byte order used to store the individual bytes of a multi-byte value, such as a 16-bit integer or a 32-bit floating-point number. Endianness plays a significant role in binary data exchange between computer systems that may use differing byte ordering. For example, if a little-endian system sends data to a big-endian system or vice versa, the receiving system needs to interpret the data correctly by reordering the bytes in the transferred data.

Some processor architectures and computer systems, such as x86, x86-64, and AMD, use little-endian. While others, like PowerPC and Motorola 68xx/68xxx, use big-endian exclusively. Some CPU architectures, such as ARM, can be configured to support both endian modes.

7.11.1 Big-endian

In a big-endian architecture, the most significant byte (MSB) of a multi-byte value is stored at the lowest memory address, and the least significant byte (LSB) is stored at the highest memory address. It is like reading a number from left to right, with the most significant digits on the left.

For example, the 16-bit integer hexadecimal value 0x1234 would be stored in memory as follows, each byte represented by two hexadecimal digits:

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  12 34                                              ....
```

7.11.2 Little-endian

In a little-endian architecture, the least significant byte (LSB) of a multi-byte value is stored at the lowest memory address, and the most significant byte (MSB) is stored at the highest memory address. It is like reading a number from right to left, with the least significant digits on the left.

For example, the same 16-bit hexadecimal integer value 0x1234 would be stored in memory as follows:

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  34 12                                              ....
```

Network protocols, file formats, and data communication standards often specify a particular endianness to ensure interoperability. When designing software or hardware that deals with multi-byte data, developers must be aware of the endianness of the target system to avoid data interpretation issues.

7.12 Text Encoding References

The following references further explain the material included in section 7.

- https://www.utf8-chartable.de/
- https://www.smashingmagazine.com/2012/06/all-about-unicode-utf8-character-sets/
- https://www.joelonsoftware.com/2003/10/08/the-absolute-minimum-every-software-developer-absolutely-positively-must-know-about-unicode-and-character-sets-no-excuses/
- https://www.i18nqa.com/debug/utf8-debug.html
- https://kermitproject.org/utf8.html

8 CONTROLLING TIMING IN FACILITIES

All digital media and computers involved in broadcast and streaming audio depend on time and frequency references to set the digital audio sample rate and to synchronize time-based events. Clock precision constrains the performance and accuracy of these items. Because all common consumer operating systems are non-realtime and computer systems use low cost realtime clocks, one of the most important challenges is keeping these systems on time. Furthermore, some of the most common operating systems do not support accurate time synchronization, which usually causes problems.

As illustrated in Figure 6, both media clock and metadata timing rely on time synchronization. All media facilities should utilize master clocks, preferably GPS locked, to synchronize all media clocks and operating system clocks. This is important to ensure that audio and metadata are synchronized between systems over the long term and that media files play at the correct speed.

Many think metadata latency is caused by metadata handling in encoders and players, when in fact the root of the problem is usually audio clock drift caused by lack of audio sample rate sync to a master clock.

8.1 International Atomic Time (TAI) and Universal Coordinated Time (UTC)

International Atomic Time (TAI) is a weighted average of around 450 atomic clocks in approximately 80 national timing laboratories around the world. The relative stability of TAI is around one part in 10^{16}.

TAI is used for the Precision Time Protocol (PTP/IEEE 1588), commonly used in financial and media applications.

TAI is not distributed for everyday timekeeping. Instead, Coordinated Universal Time (UTC) is derived from TAI by adding or subtracting an integer number of leap seconds to correct for the Earth's rotation. The number of leap seconds is chosen so that mean solar noon at the Greenwich meridian does not deviate from UTC noon by more than 0.9 seconds. As of this writing, the offset is 37 seconds. Time zones around the world are expressed using positive or negative offsets from UTC.

8.2 Comparison of various frequency/time references

Table 5 compares the performance of many commonly used frequency/time references, showing their typical frequency accuracy and the resulting maximum time error per day and per year. Units of measure have been chosen to make the numbers easily readable. Typical 10-year aging specifications of some of these references have been included for reference, although these were not used in calculating the Time Error values.

Several classes of crystal oscillators are in Table 5, including basic crystal oscillators, temperature-compensated crystal oscillators (TXCO) and oven-controlled crystal oscillators (OCXO). Additionally, the table shows rubidium atomic, cesium-beam atomic, and cesium-fountain atomic clocks for comparison. (Cesium fountain clocks are typically used as time references in international time laboratories.)

Finally, a GPS-disciplined crystal oscillator (GPSDO) is shown. Here, the time error values are pessimistic because GPS is based on TAI, and frequency errors will tend to average out in the long term.

The items in red, which show the performance of a typical computer motherboard realtime clock oscillator, are particularly significant for operators. Replying on the motherboard oscillators in individual computers can cause noticeable timing errors between computers because the accuracy of such oscillators can cause a time drift of over two seconds per day, which works out to over 13 minutes per year!

It is therefore necessary to set up such computers to synchronize to a network timeserver, such as the NIST Internet Time Service (https://www.nist.gov/pml/time-and-frequency-division/time-distribution/internet-time-service-its), so that their realtime clocks can be corrected automatically. Because a given computer’s internal clock can drift by over two seconds per day, it is usually necessary to synchronize to the timeserver several times per day. Moreover, motherboard clocks should not be used to derive audio sample rates; these should be derived from a high-precision GPS-disciplined external frequency reference, typically distributed via Wordclock, AES11 or Audio-Over-IP.

Table 5: Accuracy and time drift of various frequency/time references

Clock Type	Accuracy (ppm/ppb)	Accuracy	Aging / 10 Year	Aging / 10 Year	Time Error per day	Time Error per year
Crystal	10ppm-100ppm 20ppm 25ppm Motherboard Realtime Clock	10^{-5} - 10^{-4} $2 * 10^{-5}$ $2.5 * 10^{-5}$	10-20ppm	$10x10^{-6}$	0.864 – 8.64 secs/day 1.728 secs/day 2.16 secs/day	5.256 – 52.56 mins/year 10.512 mins/year 13.14 mins/year
TCXO	1ppm 0.1ppm	10^{-6} 10^{-7}	1ppm	$1x10^{-6}$	.0864 secs/day (86.4 ms/day) 8.64 ms/day	0.5256 mins/year (31.536 secs/year) 3.1536 secs/year
OCXO 5-10Mhz	0.02ppm (20ppb) 5ppb	$2 x 10^{-8}$ $5 x 10^{-9}$	~0.2ppm 200bpp	$0.2x10^{-6}$	1.728 ms/day 0.432 ms/day	0.63072 secs/year 0.15768 secs/year
OCXO 15-100MHz	0.5ppm (500ppb)	$5 x 10^{-7}$	~10ppb	$1x10^{-8}$	43.2 ms/day	15.768 secs/year
Rubidium Atomic	0.00001 ppm (0.01 ppb)	10^{-12}	0.005ppm 5ppb	$5x10^{-9}$	864 ns/day	315.4 ms/year
Cesium Beam	0.000001 ppm (0.001 ppb)	$1 x 10^{-13}$			86.4 ns/day	31.54 ms/year
Cesium Fountain (NPL-CsF2)	$2 x 10^{-10}$ ppm	$2 x 10^{-16}$			0.1788 ns/day (17.88 ps/day)	6.307 ns/year
GPSDO	0.001 ppm 1 ppb	$1 x 10^{-9}$			0.0864 ms/day	0.03154 secs/year

9 ID3 OVERVIEW

The ID3 tag was originally developed to facilitate inserting song-specific information directly into MPEG audio files. ID3 version 1, or ID3v1, is a fixed-size, 128 byte block of data appended to the end of MPEG audio files. It contains fixed-length fields for song name or title, artist, album, year, a comment, and an index against a pre-defined list of genres. All text fields are 8-bit ASCII, 30-bytes in length, except four bytes for the year. This structure eventually outlived its usefulness, motivating the development of ID3 version 2, or ID3v2.

ID3v2 is flexible, extensible and mainly designed for MPEG-encoded audio files. ID3v2 is a container consisting of a series of variable length blocks or frames. ID3v2 frames are easily distinguishable from audio data, so its use has expanded to digital audio transmission and streaming. Unlike ID3v1, version 2 metadata usually appears at the beginning of audio files and can accommodate tagging information of variable lengths, including image data.

The ID3v2 tag is binary. ID3 contains non-ASCII characters; therefore it must be parsed at the binary level. Software applications using ID3 frames can be written and parsed with appropriate software modules. Users can write their own or can use an available library.

ID3v2 has gone through several iterations, version 2.4 being the latest. It incorporates support for both UTF-8 and UTF-16 character formatting. ID3v2.4 UTF-8, specifically mb4, is used for HLS metadata.

All references here to ID3 below v2.4 are for historical context only.

A hex dump is a display or printout that presents each character in a file or data stream in hexadecimal format, as seen in Figure 13. It is useful for verifying that data commands are executing properly and on time. A hex dump represents a binary file and must be viewed in a hex editor, which is analogous to a text editor used to view a text file. Certain characters in a binary file cannot be viewed in a text editor.

In a hexadecimal view of data, a two-digit number represents each byte (8 bits).

Hex dumps are commonly organized into rows of 8 or 16 bytes, sometimes separated by whitespaces. Some show the hexadecimal memory address at the beginning. The right-hand side of the hex dump displays human-readable ASCII, where a dot corresponds to a non-ASCII character, including binary control characters. An example hex dump for an ID3v2.4 frame is shown in Figure 13. The dot does not show which control character is being represented; the corresponding digits in the left-hand side of the editor page do.

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  00 00 02 3F 01 49 44 33 04 00 00 00 00 04 35 54  ...?.ID3......5T
00000010  52 53 4E 00 00 00 1B 00 00 03 53 74 72 65 61 6D  RSN.......Stream
00000020  53 20 53 74 72 65 61 6D 69 6E 67 20 45 6E 63 6F  S Streaming Enco
00000030  64 65 72 00 54 52 53 4F 00 00 00 17 00 00 03 48  der.TRSO.......H
00000040  69 2D 46 69 20 49 6E 74 65 72 6E 65 74 20 53 74  i-Fi Internet St
00000050  72 65 61 6D 00 54 49 54 32 00 00 00 14 00 00 03  ream.TIT2.......
00000060  31 6B 48 7A 5F 30 64 42 46 53 5F 34 34 2E 31 6B  1kHz_0dBFS_44.1k
00000070  48 7A 00 54 50 45 31 00 00 00 0A 00 00 03 31 31  Hz.TPE1.......11
00000080  31 31 31 31 31 31 00 54 41 4C 42 00 00 00 19 00  111111.TALB.....
00000090  00 03 53 74 72 65 61 6D 53 20 50 72 65 63 69 73  ..StreamS Precis
000000A0  69 6F 6E 50 41 44 E2 84 A2 00 54 4C 45 4E 00 00  ionPADâ„¢.TLEN..
```

```
000000B0  00 08 00 00 03 31 38 36 32 30 30 00 54 43 4F 4E  .....186200.TCON
000000C0  00 00 00 05 00 00 03 4D 55 31 00 57 58 58 58 00  .......MU1.WXXX.
000000D0  00 00 4B 00 00 03 61 72 74 77 6F 72 6B 55 52 4C  ..K...artworkURL
000000E0  5F 36 34 30 78 00 68 74 74 70 73 3A 2F 2F 6C 61  _640x.https://la
000000F0  32 2E 69 6E 64 65 78 63 6F 6D 2E 63 6F 6D 2F 70  2.indexcom.com/p
00000100  6C 61 79 65 72 2F 70 61 64 2F 53 74 72 65 61 6D  layer/pad/Stream
00000110  53 5F 4C 6F 67 6F 5F 50 41 44 5F 31 2E 70 6E 67  S_Logo_PAD_1.png
00000120  54 46 4C 54 00 00 00 09 00 00 03 4D 50 47 2F 41  TFLT.......MPG/A
00000130  41 43 00 54 58 58 58 00 00 00 34 00 00 03 63 72  AC.TXXX...4...cr
00000140  62 00 53 74 72 65 61 6D 53 20 4C 69 76 65 20 48  b.StreamS Live H
00000150  54 54 50 20 45 6E 63 6F 64 65 72 20 53 65 72 76  TTP Encoder Serv
00000160  69 63 65 20 28 76 20 31 2E 35 2E 35 2E 31 32 30  ice (v 1.5.5.120
00000170  29 54 58 58 58 00 00 00 34 00 00 03 65 6E 63 00  )TXXX...4...enc.
00000180  53 74 72 65 61 6D 53 20 4C 69 76 65 20 48 54 54  StreamS Live HTT
00000190  50 20 45 6E 63 6F 64 65 72 20 53 65 72 76 69 63  P Encoder Servic
000001A0  65 20 28 76 20 31 2E 35 2E 35 2E 31 32 30 29 54  e (v 1.5.5.120)T
000001B0  58 58 58 00 00 00 21 00 00 03 64 65 76 00 4C 69  XXX...!...dev.Li
000001C0  6E 65 20 35 20 28 56 69 72 74 75 61 6C 20 41 75  ne 5 (Virtual Au
000001D0  64 69 6F 20 43 61 62 6C 65 29 54 58 58 58 00 00  dio Cable)TXXX..
000001E0  00 07 00 00 03 61 6F 74 00 32 39 54 58 58 58 00  .....aot.29TXXX.
000001F0  00 00 07 00 00 03 61 64 72 00 33 32 54 58 58 58  ......adr.32TXXX
00000200  00 00 00 0A 00 00 03 61 73 72 00 34 34 31 30 30  .......asr.44100
00000210  54 58 58 58 00 00 00 06 00 00 03 61 63 68 00 32  TXXX.......ach.2
00000220  54 58 58 58 00 00 00 1A 00 00 03 63 72 64 00 32  TXXX.......crd.2
00000230  30 31 38 31 30 31 32 20 32 31 3A 32 32 3A 31 30  0181012 21:22:10
00000240  20 55 54 43                                       UTC
```

Figure 13. Example hex dump of an ID3v2.4 frame.

9.1 ID3v2.4 metadata formats

In-band metadata is sent in the same transport as the audio data. This allows it be precisely synchronized to the audio. Elementary Stream can be synchronized to audio frame boundaries. Depending on the codec, the accuracy will be 20 or 40 msec. fMP4 is time-stamped, so synchronization can be more precise than Elementary Stream.

Out-of-band metadata, which is sometimes used for simplicity and lack of proper developer resources, is sent in a separate transport. It may be delayed in transit by a different amount of time than the audio, so out-of-band metadata cannot be synchronized. This can cause multiple problems, including title and artist appearing out-of-sync with the music, and bad splicing during ad insertion.

9.2 Issues with non-compliant HLS metadata integration

At the time of this writing, not all HLS metadata in use by streamers complies with ID3v2.4. Some of the issues with non-compliant metadata include:

- Use of Transport Stream segments for audio-only services. No standardized metadata format exists for an audio-only Transport Stream segment. Additionally, Transport Stream is considered inefficient for audio-only services due to excessive overhead.
- Metadata sent in Playlist Manifest because it is easy and text based.
  This does not require experienced developers to implement but suffers from the metadata timing problems that HLS was designed to prevent.
- Improper use and assignment of ID3v2.4 metadata frames.
- Incomplete UTF8-mb4 support.
- HLS streams that have not been updated to the new CMAF fMP4 standards. CMAF fMP4 uses a time-stamped ISO BMFF emsg box for ID3v2.4 frames.
  Many content providers and stream aggregators are unaware that the MPEG-D USAC with MPEG-D DRC (xHE-AAC™) and Lossless codecs require CMAF fMP4.
- Streams that are sourced with ICY Encoders and transpacketized at the server level.
  This causes metadata timing problems, which HLS was designed to prevent. ICY also requires a constant network connection to the server, reducing reliability. Specific HLS Segment Encoders, such as a direct HLS encoder, solve this problem.

9.3 HLS/= and DASH segment structure with in-band ID3v2.4

Table 6 shows a simplified version of the HLS/DASH Segment Structure with in-band ID3v2.4 metadata. An HLS stream with this structure can be verified using Apple iTunes/Music,[7] which is a standards-based HLS-compliant player.

[7] For example, Windows version 12.12.4.1 or Mac version 1.2.5.7 at the time of this writing.

Table 6. Simplified version of HLS and DASH segment structure with in-band ID3v2.4 (Source: StreamS/Modulation Index, LLC)

ES—Elementary Stream	fMP4 - Fragmented ISOBMFF-MP4—MPEG-DASH Compatible
HLS ES Segment **ID3v2.4 Frame** Timestamp **ID3v2.4 Frame—In-line** Stream PAD/Metadata Stream Information Audio Information Extensible Metadata Audio Gain Offset **Audio Frame** ADTS Header Audio Units Codec Loudness Control **Audio Frame** ADTS Header Audio Units Codec Loudness Control **ID3v2.4 Frame—In-line** Stream PAD/Metadata Stream Information Audio Information Extensible Metadata Audio Gain Offset **Audio Frame** ADTS Header Audio Units Codec Loudness Control **Audio Frame** ADTS Header Audio Units Codec Loudness Control **Audio Frame** ADTS Header Audio Units Codec Loudness Control **Audio Frame** ADTS Header Audio Units Codec Loudness Control REPEATS INDEFINITELY…	**HLS/DASH fMP4 Segment** **ISO emsg** Timestamp **ISO emsg ID3v2.4 Frame—Timed** Timestamp Stream PAD/Metadata Stream Information Audio Information Extensible Metadata Audio Gain Offset **ISO emsg ID3v2.4 Frame—Timed** Timestamp Stream PAD/Metadata Stream Information Audio Information Extensible Metadata Audio Gain Offset **ISO Audio Frame** Audio Units Codec Loudness Control **ISO Audio Frame** Audio Units Codec Loudness Control **ISO Audio Frame** Audio Units Codec Loudness Control **ISO Audio Frame** Audio Units Codec Loudness Control **ISO Audio Frame** Audio Units Codec Loudness Control **iso Audio Frame** Audio Units Codec Loudness Control REPEATS INDEFINITELY…

Some comments regarding HLS and DASH segment structure:

- ES and fMP4 Segments contain the same number of Audio Frames.
- ES Stream Metadata is in-line between segment audio frames.
- fMP4 Stream Metadata is based seon timestamps that reference the segment audio frames. This is more precise, as it is not limited to the granularity of frame boundaries.

- ES and fMP4 Segments must contain a timestamp frame and should contain at least one stream metadata frame.
- ES and fMP4 Segments may contain multiple synchronous in-band stream metadata frames, which compliant players are expected to support.

Normalized loudness metadata can be carried in both the stream metadata and the codec metadata.

- Stream metadata will accommodate normalized level control for audio codecs that do not support Codec Metadata, such as AAC.
- Codec metadata will accommodate normalized level control for audio decoders that do support codec metadata, such as xHE-AAC.

Loudness metadata can be carried in the stream metadata and/or the codec metadata. Where codec metadata is not supported or reliable, stream metadata should be used to allow for normalized level control. When codec metadata is supported and mandated, as with MPEG-D USAC (ISO/IEC 23003-3) when carried in CMAF (ISO/IEC 23000-10), or in specific implementations of MPEG-D USAC such as xHE-AAC™, the audio decoders themselves will provide normalized level control.

9.4 Metadata transport

Some codecs cannot convey all metadata that users need, and even if a given codec supports such metadata, not all players respond to it. As an alternative or backup to codec metadata, user-required metadata can be included in the content's ID3 frame and parsed with JavaScript.

10 METADATA FOR HD RADIO

HD Radio is an in-band/on-channel (IBOC) digital broadcasting technology that provides significantly more metadata capability than the RDS metadata used for analog FM broadcasts.[8]

HD Radio technology can be used in both FM and AM transmission systems. In the AM band, stations transmitting HD Radio signals have two options: mode MP1 with 36 kbps of data throughput (hybrid digital radio with both analog and digital components,[9] not to be confused with hybrid radio that includes internet-delivered content) or MA3, an all-digital signal with 40 kbps of data throughput. FM stations have more options—hybrid mode (MP1) 98 kbps or extended hybrid modes (MP2 through MP11) with 110-148 kbps of throughput. In either AM or FM, the data content can be divided into multiple audio channels known as "multicast channels" with designations such as HD2, HD3, etc. (up to HD8).

Figure 14 shows how audio and metadata flow from playout systems for three program streams (FM/HD1, HD2, and HD3) to an FM HD Radio transmission system. While essential to the operation of the system, GPS-disciplined word-clock connections are not shown.

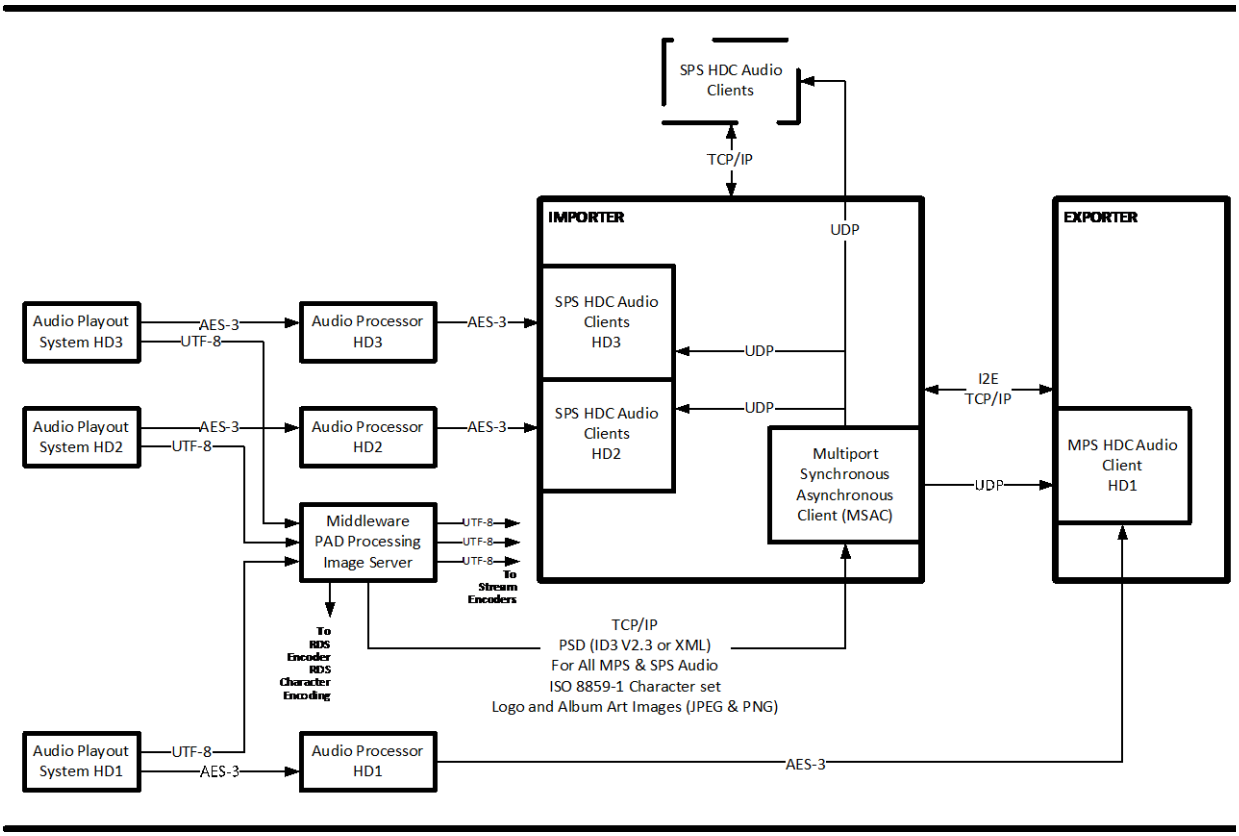


Figure 14. HD Radio audio and metadata flow diagram.

HD Radio broadcasts offer a variety of data services. One of the more widely known is Program Service Data (PSD), which supports a broad set of categories that describe the programming content. The Title field can describe the name of a song, the topic of a talk show, an advertisement, or an announcement.

[8] "HD Radio™" is a trademark of Xperi Inc.
[9] In this context "MP" stands for "Main Program" and is unrelated to the "MP" used in codec names like "MP3." In the figure, "SPS" denotes "supplemental program stream" and "HDC" is the proprietary audio codec for HD Radio. "I2E" denotes "importer-to-exporter link."

Commonly used PSD fields include the following:

- Title
- Artist
- Album
- Genre
- Comment
- Commercial
- Reference identifiers

The PSD typically originates from a studio automation system or any other computing resource where program audio originates and is multiplexed with the audio streams. Some essential requirements for HD Radio PSD are:

- Program audio and associated data shall be transmitted synchronously so receivers can acquire correlated audio and data simultaneously.
- PSD messages shall not exceed 1024 bytes.

Some key considerations for PSD are:

- PSD messages are continuously transmitted, with the most recent message transmitted repeatedly.
- PSD providers send a new PSD message when the PSD content has changed.

PSD is formatted for the HD Radio system using a subset of the standard called ID3v2.3 (which is not compatible with ID3v2.4). The ID3v2.3 general structure is as follows:

- The complete ID3 message is called an ID3 tag.
- ID3 tags contain one or more content types referred to as frames. Frames contain individual segment information (e.g., song artist, title, etc.). Each frame has a four-character identifier. For example, the commercial frame is identified as COMR. The HD Radio software automatically handles ID3 framing.
- Within frames, sub-elements called fields can exist. Fields further categorize the information within a frame. For example, the commercial frame has a field to specify the sale price.
- HD Radio accepts only ISO-8859-1 characters for Main (MPS) and Supplemental (SPS) Program Service Data. This should be enforced in either the playout system or the metadata bridge software. Metadata bridge software is preferred, as this will allow the playout system to provide richer metadata for other media.

HD Radio Artist Experience is the synchronous transmission, delivery, and display of images related to the specific audio segment on the receiver. Cover art images may also include artist photos, slideshows, or other images related to the song or audio content being played. This feature can also display commercial images related to an advertisement segment.

The image is closely synchronized with the song or audio being played and displayed one at a time. If the cover art or primary image is unavailable, the station logo or other default image is displayed on the receiver.

The images have a maximum resolution of 200x200 pixels and a maximum file size of 24 kilobytes.

The image support data client is optimized to make use of the available bandwidth based on the following factors:

- Size of image
- Image repeat rate
- Image data transfer rate

Additional information is available on the HD Radio website at
https://hdradio.com/broadcasters/engineering-support/artist-experience/

11 INTERSTITIAL CONTENT INSERTION

Interstitial content such as commercials can be inserted from a local source or external network(s). Precise timing is required, which requires synchronizing all program sources (program content and all interstitials) to a network timeserver. When used with the HLS or MPEG-DASH streaming protocols, this allows smooth and seamless on-time ad insertion (e.g. DAI, SSAI).

11.1 SCTE-35 - Digital Program Insertion Cueing Message

SCTE 35 (2022b) is the core signaling standard for advertising at the network level. It provides targeted program and distribution control (e.g., blackouts) of content for content providers and content distributors. SCTE 35 signals, otherwise known as “triggers”, are usually generated by the playout system and require specific encoder support. They can be used to identify advertising breaks, advertising content, and programming content (e.g., specific programs and chapters within a program). Triggers work with direct ad insertion for live streams; these are widely used in television as well as radio. They consist of metadata used to mark and signal information related to a certain timestamp or time range in a stream.

SCTE-35 is a joint ANSI/Society of Cable and Telecommunications Engineers standard. The full standard name is “Digital Program Insertion Cueing Message for Cable.” SCTE 35 complements other Standards to complete the eco-systems:

- SCTE 130-3 is used to support alternate content decisions (advertising, blackouts, stream switching) for live and time shifted delivery;
- SCTE 214-1 defines how SCTE 35 is carried in MPEG-DASH. [SCTE 224] (ESNI) is used to pass event and policy information from provider or other systems to communicate distribution control instructions;
- SCTE 35 is not the only method available to signal ad insertion in streaming. While other methods are in use, they don’t have the benefit of SCTE 35 standardization.

In general, ad insertion poses several technical challenges. Providers require up-to-date IP tables to send targeted advertising to the correct destinations. The inserted content must have loudness that is consistent with the loudness of the network content.

The recommended practices for SCTE 35 are contained in [SCTE 67] “Recommended Practice for Digital Program Insertion for Cable”. The standard supports delivery of events, frame accurate or non-frame accurate, and associated descriptive data in MPEG-DASH and HLS. This standard supports the splicing of content (MPEG-2 transport streams, MPEG-DASH, etc.) for the purpose of digital program insertion, which includes advertisement insertion and insertion of other content types. The standard defines an in-stream messaging mechanism to signal splicing and insertion opportunities. As such, the standard does not specify the insertion method used or constraints applied to the content being inserted, nor does it address constraints placed on insertion devices.

This standard specifies a technique for carrying notification of upcoming points and other timing information in the transport stream. A splice information table is defined for notifying downstream devices of splice events, such as a network break or return from a network break.

For HLS, manipulation of an HLS m3u8 manifest is used to provide seamless ad insertion. The manifest is modified to include targeted ads prior to delivery to the player or the manifest is modified at the player before delivery to the device’s video playback engine. These mechanisms allow for seamless playback without buffering or other interruptions in the playback experience. HLS m3u8 manifest manipulation can be done on a server or on a client (for example, to implement companion ads). Client-side ad insertion typically would need a secure player implementation to ensure the ad segments play out correctly.

The SCTE 35 standard presents two alternatives for SCTE 35 messages that are carried in HLS. The recommended approach is based on [HLS-TMD] and is clarified in Section 12.2.1 of the Standard. The legacy approach is specified in Section 12.2.2. Both client- and server-side methods can be used in the same manifest. The implementer must provide appropriate markings for the targeted devices and players.

For MPEG-DASH, the splice information table is carried in the DASH MPD (See [SCTE 214-1]) or in media segments (see [SCTE 214-2] and [SCTE 214-3]).

11.2 HLS Interstitial Content Insertion Flow Diagram and Discussion

Millisecond-accurate insertion of interstitial content into HLS and MPEG-DASH at the encoder side requires the player's splicing content and metadata within HLS or MPEG-DASH frames, not just on frame boundaries, which is where the encoder inserts the metadata.

For splicing to occur successfully, files containing the ads to be inserted must be prepared by encoding them with the same codec, sample rate, and bit depth used for the main program. The metadata must maintain time alignment with the content during file encoding, which typically depends on synchronizing both source and destination to a centralized network timeserver. The streaming server handles the intricate details of glitch-free insertion of the ad content and metadata at precisely the desired time by inserting the contents of the previously-encoded ad files into the final stream.

The diagrams below show ad insertion workflow and segment splicing for HLS and compare it to the workflow for ICY, whose metadata timing accuracy is far poorer than that offered by HLS and MPEG-DASH:

- Figure 15. HLS fMP4 interstitial content insertion flow diagram.
- Figure 16. ICY interstitial content insertion flow diagram.

In Figure 15 and Figure 16, "DISCON" refers to "discontinuity," where DISCON is the EXT tag in the HLS m3u8 manifest file.

One of the biggest challenges in HLS content insertion is slicing the HLS segments appropriately while maintaining metadata synchronization. Elementary Stream (ES) and fragmented MP4 (fMP4) segment file structures are completely different. ES segments are sliced ADTS frames at frame boundaries, and are limited to a timing accuracy of 20-40 milliseconds depending on frame size. fMP4 segments are sliced ISO-BMFF file segments at frame boundaries, but with timestamps that instruct the player where to insert the content within the frame. fMP4 timing accuracy can be better than 1 millisecond.

As of this writing, fMP4 remain largely unsupported at the CDN level because CDNs and content providers typically lack the necessary developer resources. This complexity can impede correct HLS deployment, although a direct HLS encoder that handles the details of the segment splicing can remove a great deal of the burden that would otherwise fall on the developers working for CDNs and content providers.

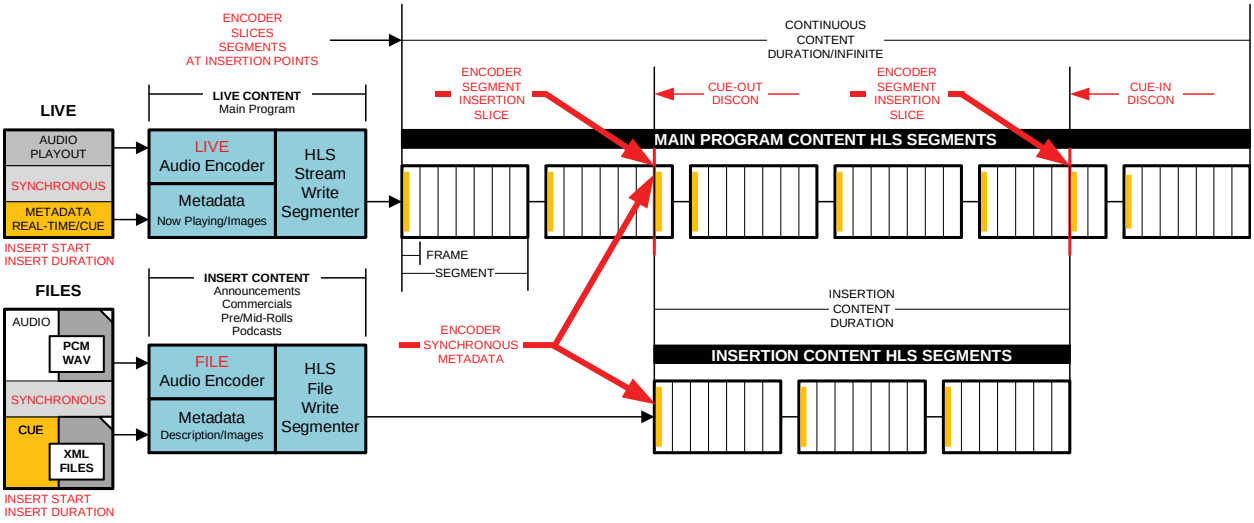


Figure 15. HLS fMP4 interstitial content insertion flow diagram.
(Source: StreamS/Modulation Index/Greg Ogonowski)

11.3 ICY Interstitial Content Insertion Flow Diagram and Discussion

The ICY (Icecast or SHOUTcast v1) streaming encoder output contains two elements: the audio bitstream and the metadata, which is transmitted out-of-band. The ICY server is responsible for assembling the audio bitstream together with the in-line metadata sent at a server-defined metadata interval. It is completely asynchronous and causes a random, unpredictable metadata time offset at the player with reference to the real-time metadata. Hence, it is impossible to use the ICY streaming protocol for any kind of precisely timed metadata, especially precision content control.

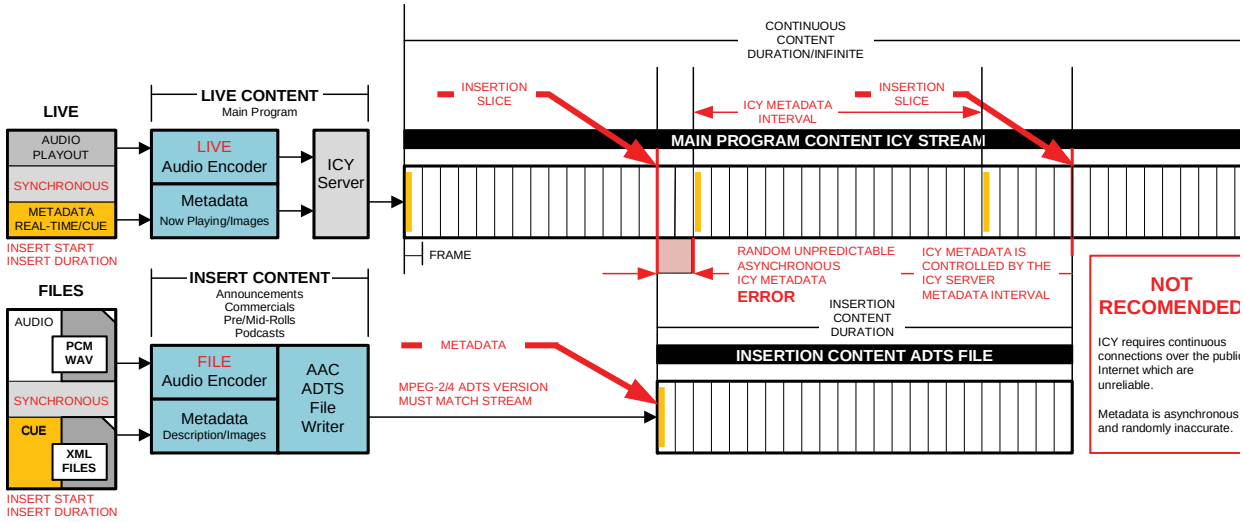


Figure 16. ICY interstitial content insertion flow diagram. (Source: StreamS/Modulation Index/Greg Ogonowski)

12 OTHER RELEVANT STANDARDS

12.1 AES3

AES3 is a standard published by the Audio Engineering Society as AES3-2009 (r2019), for the exchange of digital audio signals between professional audio devices. An AES3 signal can carry two channels of pulse-code-modulated digital audio over several transmission media including balanced lines, unbalanced lines, and optical fiber. AES3 has been incorporated into the International Electrotechnical Commission's standard IEC 60958 and is available in a consumer-grade variant known as S/PDIF (IEC 60958 type II).

AES3 comes in two variants. IEC 60958 type I connections use balanced, three-conductor, 110-ohm twisted pair cabling with XLR connectors (shown in Figure 17). Type I connections are most often used in professional installations and are considered the standard connector for AES3. The AES-3id standard defines a 75-ohm BNC electrical variant of AES3 and typically utilizes BNC-type connectors. This uses the same cabling, patching and infrastructure as analogue or digital video, and is thus common in the broadcast industry. Being correctly impedance-controlled (including connectors), AES3id is capable of longer error-free cable runs than the Type-1 version.

Figure 17. IEC 60958 type 1 connections. (Source: Wikipedia)

AES3 supports metadata, including basic control data: sample rate, compression, pre-emphasis, if the audio stream is stereo, mono or some other combination, audio word length, additional sample rate information not representable by the basic sample rate in Byte 0, channel origin, channel destination, and channel status word reliability indication. AES metadata includes user bits, which are available for customized applications such as inclusion of now-playing information and/or loudness data.

SMPTE timecode data can be embedded within AES3 signals. It can be used for synchronization and for logging and identifying audio content. It is embedded as a 32-bit binary word in bytes 18 to 21 of the channel status data The AES11 standard provides information on the synchronization of digital audio signals and recommends using an AES3 signal to distribute audio clocks within a facility. The AES52 standard describes how to insert unique identifiers into an AES3 bit stream.

12.2 USAC/xHE-AAC™

The Unified Speech and Audio Codec (USAC) aggregated the existing AAC codec family and introduced new technology. It combines and improves upon HE-AAC for music and generic audio and AMR-WB+ for speech. USAC further builds on the technologies in MP3 and AAC and takes these one step further. It includes all the essential components of its predecessors and improves them.

As of this writing all popular operating systems include eight-channel AAC decoders, making surround streaming easy for users.

AAC supports MPEG-D metadata, standardized in ISO/IEC 23003-4:2020 as amended. This includes Dynamic Range Control, True Peak Level, Sample Peak Level, Anchor Loudness, and Program Loudness metadata. Section 8.2.1.1 of ANSI/CTA-2075 describes how player devices should use this metadata when available.

Signaling and transport of MPEG-D USAC is similar to MPEG-4 HE-AACv2, but a USAC decoder unambiguously determines its configuration at startup, and there is no delay in SBR or PS. Extended HE-AAC decoders will decode AAC-LC, HE-AACv1, HE-AACv2, and MPEG-D USAC bitstreams.

First introduced within the USAC standard, Extended HE-AAC adds a new set of encoding tools to the HE-AACv1 and v2 audio codecs. It significantly improves the audio quality of music and speech particularly at very low bitrates of 8 kbit/s to 64 kbit/s and is compatible with HE-AAC streams. Extended HE-AAC outperforms dedicated speech and general audio coding schemes and bridges the gap between both worlds, providing consistent high-quality audio for all signal types. Accordingly, Extended HE-AAC can improve the quality of existing low bitrate services or more audio channels can be transmitted at a given bitrate.

USAC preserves the same overall structure of HE-AACv2. The core coder consists of an AAC based transform coder, enhanced by ACELP speech coding technology AMR-WB+. An enhanced Spectral Band Replication (SBR) tool, eSBR, handles high frequencies, while MPEG Surround supplies parametric stereo coding.

xHE-AAC™ is Fraunhofer's trademark covering its implementation of USAC. It mandates inclusion of loudness and dynamic range control metadata.

For a discussion of how AAC is conveyed in audio media files, see section 13.2 (AAC).

12.3 XML (Extensible Markup Language)

XML is a data format that has wide applications throughout the field of information technology and supports user-defined fields for data exchange. It is the preferred method of delivering PAD/metadata to a live streaming encoder. It is standards-based and supports UTF-8. It is extensible, allowing users to define fields that are needed to support their applications. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications, all of them free open standards, define XML.

Unlike text files, which need a proprietary format definition and delimiters, XML is self-delimiting. There are many parsers available, including JavaScript, to put PAD/metadata in an HTML document for websites.

White spaces are optional in XML documents and are sometimes added to facilitate readability. XML documents can contain international characters, like Norwegian øæå or French êèé. To avoid errors, specify the encoding used, or save the XML files as UTF-8, which is the default character encoding for XML documents.

XML files must be delimited by an opening and closing tag and cannot be concatenated. Sending XML data via TCP-IP must be done carefully. Persistent TCP-IP connections can cause XML files to be concatenated and should be avoided. UDP-IP has no such issue.

13 AUDIO MEDIA FILES

This section describes metadata that audio media files can carry. Note that metadata for streaming and files are not the same. Although there may be some similarity, their purpose, implementation, and operation are quite different. Media file types discussed in this section include the following:

- MP3
- AAC
- USAC
- WAV
- AIFF
- ALAC
- FLAC
- Audio CD file extraction (CD ripping)

Audio media files should be viewed with a hex editor (see Figure 13 for an example of a hex dump). Hex editors show the entire file contents without parsing errors, which can sabotage software development and metadata debugging.

Useful hex editors include:

- Microsoft Windows
  HxD
  https://mh-nexus.de/en/hxd/
- Apple macOS
- Hex Fiend
  https://hexfiend.com/

13.1 MP3

There is no standard file extension for RAW MP3 because it requires a container to be useful. Unlike AAC, MP3 frames are almost always formatted in standard MPEG ADTS bitstreams. The MP3 file format specifies this. ADTS MP3 is a containerless transport stream format, not a file format. The ADTS MP3 bitstream file format is the equivalent of ADTS AAC bitstream file format except for different audio coding and no AAC file extension.

Although it is possible to use RAW MP3 frames inside an MP4 container, it is extremely rare in practice. Few MP3 encoders can produce RAW MP3 frames, so ADTS MP3 is commonly used even though it is inefficient because redundant headers are constantly sent in the bitstream.

See [13] and [14] for additional information.

13.2 AAC

Unlike MP3, AAC frames are commonly used in several different bitstream formats. This can be confusing to developers and cause implementation mistakes. There are MPEG-2 and MPEG-4 versions of AAC, and they should not be mixed within a given application such as streaming and content/commercial insertion because there is no guarantee that the player can change bitstream modes in the middle of a stream.

AAC files are rarely used without a container, but AAC files are often described incorrectly as RAW AAC. RAW AAC files are commonly put in MP4/M4A containers and use iTMS (iTunes Music Store) metadata. Although ID3 can be used within MP4/M4A containers, iTMS metadata has gained popularity because of its penetration in the consumer space.

M4S file segment containers for fMP4 HLS/DASH streaming also use RAW AAC, except that the metadata is in a time-stamped emsg ID3 ISO box and can contain as many ID3 frames as the program material needs. Player clients are responsible for reading and displaying all ID3 frames in sync with the audio. M4S files require MP4 init files, which contain the codec header information.

ADTS AAC is a containerless transport stream format, not a file format. ADTS AAC is less efficient than RAW because redundant headers are constantly sent in the bitstream. HLS ES (Elementary Stream or Packed Audio) use segmented ADTS AAC files. Metadata in HLS ES is ID3v2.4, and segments can contain as many ID3 frames as the program material needs.

ADTS AAC is used for ICY streaming, which employs asynchronous, proprietary metadata, not ID3. ICY streaming should be avoided because its metadata is never on time and is thus inadequate for content/commercial insertion and now-playing displays. Relevant standards documents include:

ISO/IEC 13818-7:2006
Information technology — Generic coding of moving pictures and associated audio information — Part 7: Advanced Audio Coding (AAC)
https://www.iso.org/standard/43345.html

ISO/IEC 14496-12:2015
Information technology — Coding of audio-visual objects — Part 12: ISO base media file format
https://www.iso.org/standard/68960.html

ISO/IEC 23000-19:2020
Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media
https://www.iso.org/standard/79106.html

13.3 USAC

USAC and its Extended HE-AAC profile mostly uses ISO/MPEG-4 containers, including fMP4 HLS/MPEG-DASH segmented streaming. It can also use Low Overhead Audio Stream (LOAS), but support for LOAS is not universal and there is no metadata standard for it.

13.4 WAV

WAV is a container format based on extensible RIFF, developed by Microsoft and IBM. WAV usually contains PCM uncompressed audio data in various formats, but not always. It can support up to 65,535 audio channels.

Although not widely known, the WAV container can incorporate compressed audio formats, including MP3.

WAV files can contain either native LIST INFO metadata or other formats such as extensible ID3, or Broadcast WAV Format (BWF). Two metadata chunks commonly used in broadcast RIFF/WAV (.wav) audio files, referred to as Broadcast Wave Format, are “bext” (EBU-TECH 3285) and “CART” (AES Standard AES46-2002). Both contain fixed-length fields for title, artist, origination, date and various timing parameters to facilitate preserving timestamps across systems.

Native tagging LIST INFO is limited to ASCII characters. There is no UTF-8 support. (It has been attempted, but it is very non-standard.) Microsoft Windows Explorer supports .wav file metadata as ASCII, not UTF-8, with limited fields.

It is difficult to find accurate and correct information regarding WAV file metadata. Online, there is a plethora of erroneous, misinformation regarding WAV file tagging, including false claims that metadata is not supported. One source for information on the WAV file is here:

Audio File Format Specifications
WAVE
http://www-mmsp.ece.mcgill.ca/Documents/AudioFormats/WAVE/WAVE.html

13.5 AIFF

AIF/AIFF is a container format based on extensible IFF. It conforms to the EA IFF 85 Standard for Interchange Format Files developed by Electronic Arts. The AIFF container usually contains PCM uncompressed audio data in various formats, but not always—it may contain compressed audio, sometimes AIFF-C or AIFC. .

AIFF files can contain either native chunk metadata or other formats such as extensible ID3. Native metadata is limited to only ASCII characters. However, custom metadata chunks can be used, including ID3.

It is difficult to find accurate and correct information regarding AIFF file metadata. Online, there is a plethora of erroneous misinformation regarding AIFF file tagging, including false claims that metadata is not supported. One source for information on the AIFF file is here:

Audio File Format Specifications
AIFF/AIFF-C
http://www-mmsp.ece.mcgill.ca/Documents/AudioFormats/AIFF/AIFF.html

13.6 ALAC

Apple Lossless Codec (ALAC) is an open-source and royalty-free coding format developed by Apple. ALAC supports up to 8 channels of audio at 16, 20, 24 and 32 bit depth with a maximum sampling rate of 384 kHz. ALAC data is frequently stored within an MP4 container with the filename extension .m4a. This extension is also used by Apple for lossy AAC audio data in an MP4 container (same container, different audio encoding). When ALAC is a file, it goes into a M4A container. When it is in an fMP4 HLS stream, it goes into a M4S container. One source for information on the ALAC file format is here:

Apple Lossless Audio Codec
https://macosforge.github.io/alac/

13.7 FLAC

Free Lossless Audio Codec (FLAC) is open-source. As a lossless codec, its output is identical to its input, like a zipped file. FLAC supports up to eight audio channels for surround applications.

FLAC audio files use UTF-8 Vorbis Comments for metadata. When FLAC is used for fMP4 HLS/DASH streaming, the FLAC frames must be inside an ISO MP4/M4S container. Metadata is in a time-stamped emsg ID3 ISO box and can contain as many ID3 frames as the program material needs. Player clients are responsible for parsing and displaying all ID3 frames in sync with the audio. One source for information on FLAC is here:

FLAC
https://xiph.org/flac/

13.8 Audio Compact Disc File Extraction—CD Ripping

Some CDs contain useful metadata such as title and artist in the form of CD-Text; unfortunately, many CDs do not contain this information. CD rip metadata can be extracted from CD text if it is present on the CD being ripped. CD-Text is an extension of the Red Book Compact Disc specifications standard for audio CDs. It allows storage of additional information (e.g., album name, song name, and artist name) on a standards-compliant audio CD. The specification for CD-Text was included in the Multi-Media Commands Set 3 R01 (MMC-3) standard, released in September 1996. It was also added to new revisions of the Red Book (CD standard). The actual text is stored in a format compatible with Interactive Text Transmission System (ITTS), defined in the IEC 61866 standard.

Good audio CD file extraction software should properly tag the resulting audio files. This will allow standards-based playout systems to read and import this metadata into the playout system database correctly. Even if the source CD lacks CD Text metadata, CD-ripper software can often deduce its identity

via online services such as GD3, Gracenote, MusicBrainz, Discogs, and Freedb, which can supply metadata for file tagging. CD ripper software should be carefully selected and tested, as not all applications extract and tag information correctly. To preserve audio quality, the software should be configured to output audio in linear PCM format.

One example of a CD ripping program that has metadata support is “Exact Audio Copy” and may be found at https://www.exactaudiocopy.de/en/.

14 PLAYOUT SYSTEMS

Playout systems are responsible for outputting high-quality audio and synchronous realtime metadata. A simplified workflow for a typical playout system is as follows (see Figure 18:

1. Rip CDs and obtain valid tag metadata, usually from an external database.
2. Ingest audio files and metadata into the playout system.
3. Schedule/load audio playlist/program log.
4. Schedule/load other events and interstitials.
5. Play audio.
6. Result:
    a. Hear audio.
    b. View program associated metadata.

There are no standards for this playout system-generated metadata other than standard data types, which many systems don't follow. Some playout systems use a proprietary metadata format, which can be converted using third-party utilities. Furthermore, many of these systems lack Unicode support and have fixed-length record databases with insufficient field lengths. Some only support writing files that contain the Now Playing information. These file creation and retrieval implementations can have many security and permissions issues, as well as excessive latency.

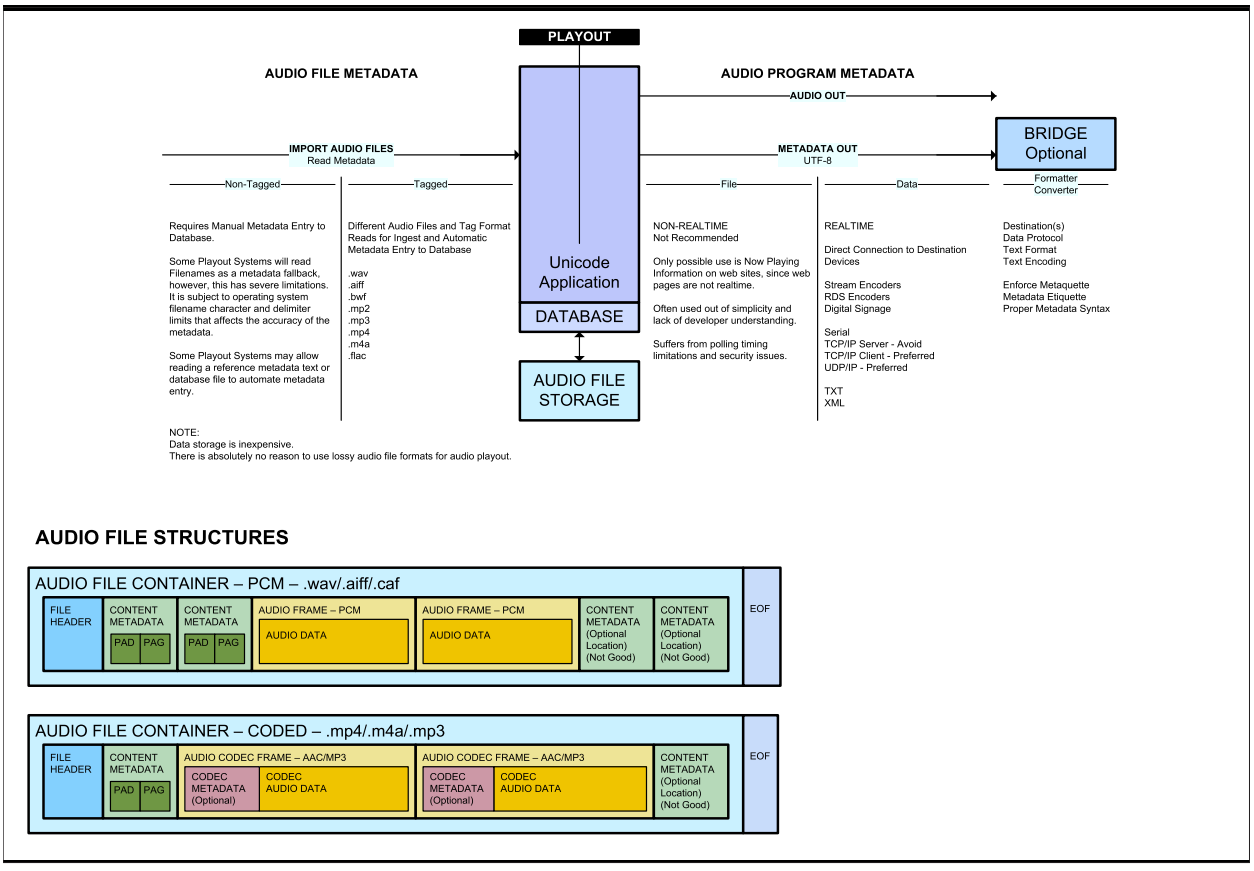


Figure 18. Typical metadata data flow in content provider's facility (playout system shown in top center of diagram).

Thus, to achieve interoperability and systems integration compatibility, use standards-based file formats and protocols. This includes standard audio file formats, text file formats, metadata output protocols and templates, and database protocols. Metadata output templates, preferably standards-based using XSML

or XML, should be used to achieve maximum compatibility with intended destinations. Optionally, the playout metadata may feed PAD/metadata middleware for extra text processing and/or insertion.

The highest-performance systems are database-driven. The database stores all audio file inventory, including metadata, playlist/program logs, and event schedules. It also allows multi-user access, which is a requirement for large systems. Databases also allow managing audio inventory programmatically without having to edit audio media files, which can increase operational efficiency. If audio files ingested into the playout system are tagged with metadata, the playout system reads the metadata upon ingest and updates the database accordingly.

Full-featured playout systems allow the use of a cue list to send multiple metadata fields that appear at specified times during playout of a single audio element. This is useful for commercials and long form programming such as music countdown shows. This metadata can also be used for logging and affidavits.

By default, playout systems should not automatically change audio or metadata files. Any modifications should be done externally with specific utilities. Upon file ingest to the system and database, the file metadata should be read, if present, and inserted into the database. If there are typos in the original file metadata, they should be fixed in the playout system database, and possibly in the file with external utilities. Fixing the audio file itself is entirely optional, as some playout systems ingest audio files with no metadata at all, in which case the metadata must be ingested separately from another database or text file.

Although beyond the scope of this document, playout audio sample rate conversion accuracy is also important, especially since the majority of audio sources are 44.1kHz and may be playing through 48 kHz audio devices, such as AES-3 or AES-67.

The playout system database standard is Structured Query Language (SQL) and there are several choices. Using SQL allows maximum flexibility, reliability and scalability. SQL versions appropriate for streaming metadata support the utf8mb4 character set, which is essential to achieve the highest compatibility with modern text strings because they may include emoji and characters used in languages other than English. It is important that the database used supports utf8mb4; as of this writing, not all have been upgraded.

Using files for metadata is highly discouraged because of latency and security issues. The only allowable exception to this is when metadata needs to be displayed on a web page. There may be no other option than to transfer a file, although this can be overcome by using PHP sockets and accepting a realtime data stream exactly like a streaming encoder.

The least reliable way to ingest metadata into a playout system is by reading the filename. Certain characters in filenames are reserved for operating systems and therefore cannot be used. This constrains filename-based metadata. Furthermore, the traditional artist-title separation character (the hyphen) will cause trouble if either artist and/or title contain a hyphen.

For a list of forbidden characters in filenames and examples of how this limits how Title and Artist can be correctly encoded, see section 6.3.

14.1 Example Playout System Metadata Templates

Some playout systems are template-based and the examples below are of XML outputs with various fields representing metadata.[10] The XML is encoded in UTF-8 and this is declared in the header. Note that for each example below, it is important to declare UTF-8 in the XML Declaration Header, even though UTF-8 is mandatory for XML.

[10] The figures in this section have been provided courtesy of StreamS/Modulation Index, LLC/Greg Ogonowski

14.1.1 Playout System to Streaming Encoder

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[Artist Variable Goes Here %artist%]]></artist>
        <title><![CDATA[Title Variable Goes Here %title%]]></title>
        <image1>http://www.domain.com/img/filename.png %image%</image1>
        <image2>http://www.domain.com/img/filename.png %image%</image2>
        <image3>http://www.domain.com/img/filename.png %image%</image3>
        <trackid>Track ID Variable Goes Here</trackid>
        <duration>Duration Variable Goes Here</duration>
        <album><![CDATA[Album Name Variable Goes Here %album%]]></album>
        <year>Year Variable Goes Here %year%</year>
        <accompaniment>Accompaniment Variable Goes Here</accompaniment>
        <group>Group Variable Goes Here</group>
        <composer>Composer Variable Goes Here</composer>
        <isrc>US-S1Z-15-00001</isrc>
        <category>Category Variable Goes Here</category>
        <audio_gain>1.000</audio_gain>
        <ext>
                <item>
                        <key>ext_01</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_02</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_03</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_04</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_05</key>
                        <value>0</value>
                </item>
                <item>
```

```
                        <key>ext_06</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_07</key>
                        <value>0</value>
                </item>
                <item>
                        <key>ext_08</key>
                        <value>0</value>
                </item>
        </ext>
</Event>
```

It is important to declare UTF-8 in the XML Declaration Header, even though UTF-8 is mandatory for XML.

14.1.2 Content Insertion

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[Artist Goes Here]]></artist>
        <title><![CDATA[Title Goes Here]]></title>
        <image1>http://www.domain.com/img/filename.png</image1>
        <cue type="Commercial" duration="00:00:00" />
</Event>
```

It is important to declare UTF-8 in the XML Declaration Header, even though UTF-8 is mandatory for XML.

14.1.3 Emoticon/Emoji Usage in XML

Using emoticons/emoji is complicated. Achieving all the features they have to offer requires attention to detail.

Emoticon/emoji characters may be used in one of three ways in XML. It is up to the stream encoder and decoder to support them properly.

Some characters require Sequences, which are two or more codes concatenated with no space between them. Flags are an example.

1. Unicode Hex Code - HTML Format `&#x1F1FA;&#x1F1F8;`
   This should render to: 🇺🇸

For emoticons/emojis, the Variation Selector tells the system rendering the character whether it should be treated as text or as an image, which could have additional properties, like color or animation. For emoji there are two different variation selectors that can be applied, U+FE0E and U+FE0F. U+FE0E specifies that the emoticon/emoji should be presented as text. U+FE0F specifies that it should be presented as an image, with color and possible animation.

Without the variation selector the character still renders fine, but there's no additional information about presentation.

Example: Sparkles
1. HTML Code `&#10024;`
2. Unicode Hex Code - HTML Format `&#x2728;`
3. Actual Character within XML CDATA ✨

The following links provide more details:

- https://unicode.org/emoji/charts/
- https://unicode.org/Public/emoji/15.0/emoji-test.txt
- https://unicode.org/Public/emoji/15.0/emoji-sequences.txt
- https://unicode.org/Public/emoji/15.0/emoji-zwj-sequences.txt
- https://unicode.org/Public/15.0.0/ucd/emoji/emoji-data.txt
- https://unicode.org/Public/15.0.0/ucd/emoji/emoji-variation-sequences.txt
- http://unicode.org/emoji/charts/emoji-variants.html

Note that as of this writing, international flags are not supported in the Edge, Chrome and Office applications for Microsoft Windows. Instead, they are shown as 2-digit ISO Country Codes. For example:
US
International flags are supported in Firefox and other operating systems and devices.
Moreover, Windows is also behind the current Unicode Emoticon version. Many emoticons/emojis are missing.

```
<?xml version="1.0" encoding="UTF-8"?>
<Event>
        <artist><![CDATA[Artist Variable Goes Here %artist%]]></artist>
        <title><![CDATA[Title Variable Goes Here %title%]]></title>
        <album><![CDATA[🅱️BIG 8 Radio]]> $year$</album>
        <album>&#127921; BIG 8 Radio $year$</album>
        <album>&#10024;<![CDATA[✨ Bubblegum Galore! $year$ ✨]]></album>
        <album>&#x1F1FA;&#x1F1F8;<![CDATA[🇺🇸USA Flag $year$ 🇺🇸]]></album>
        <image1>http://www.domain.com/img/filename.png</image1 %image%>
</Event>
```

It is important to declare UTF-8 in the XML Declaration Header, even though UTF-8 is mandatory for XML.

Simplified Emoji Usage Test:

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Emoticon Usage Test</title>
</head>
<body>

<h2>Emoticon/Emoji Test</h2>
<br>
The Infamous Smiley Face<br>
utf8mb4<br>
😀
&#x1F600;
&#128512;
<br>
<br>
Violin<br>
utf8mb4<br>
🎻
&#x1F3BB;
&#127931;
```

```
<br>
<br>
Sparkles<br>
utf8mb3<br>
✨
&#x2728;
&#10024;
<br>
<br>
Copyright Sign<br>
utf8mb2<br>
©
&#x00A9;
&#169;
<br>
<br>
Check Mark<br>
&#x2714;
<br>
Check Mark – with Variation Selector<br>
&#x2714;&#xfe0f;
<br>
<br>
USA Flag<br>
&#x1F1FA;&#x1F1F8;
<br>
&#x1F1FA;&#x1F1F8;&#xfe0f;
<br>

</body>
</html>
```

It is important to declare UTF-8 in the HTML Meta Tag. Some servers and devices may require a BOM at the front of the document.

14.1.4 Hyperlink Metadata

To direct users to web pages, clickable links can be sent via metadata. This can include items such as home pages, special offers, and coupon pages. The clickable elements can be either text or graphic images. Scannable QR Codes can also be used for these links.

Security *must* be observed when including clickable links in metadata.

The example below shows how to make a clickable HTTP link.

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Emoticon Test</title>
</head>
<body>

<div id="logo1">
<a href="https://www.indexcom.com" target="_new" >
<img onload="this.style.opacity='1'" src="img/Logo_StreamS.svg"  />
</a>
</div>

</body>
</html>
```

15 LEGAL CONSIDERATIONS

The following presents a brief overview of some of the legal and copyright material issues associated with streaming and metadata.

15.1 Codec Licensing

While the patents covering MP3 are now expired and other codecs like Opus are royalty-free, some audio codecs still require licensing. As of this writing, there are several streaming encoders available that incorporate unlicensed codecs. It is wise to verify that the vendor of the streaming encoder in use is properly licensed.

15.2 Copyrighted Material

Along with the music performance reports that must be submitted to satisfy the requirements of **SoundExchange**, there may be other copyright licenses that need to be obtained. Some of these include:

- Album Cover Art
- Artist Images
- Lyrics

This information, when supplied by third-party metadata services, must be fully licensed by the service provider.

15.3 Audience Analytics

Audience analytics are the lifeblood of broadcasting and streaming, providing information about audience use of the broadcast and/or stream that allows ad availabilities to be priced fairly. However, there are several potential sources of error in analytics that can cause the number of listeners, their demographics, and/or their physical locations to be misreported. Some of these include:

- Reporting that may be inadequately audited for errors.
- Stale IP tables that report false listener locations.
- Multiple players sharing one IP address or domain.
- Players or receivers connected to streams or broadcasts when no one is actively listening, an issue that devices such as the Nielsen **Portable People Meter** are intended to address.
- VPN connections that mask certain analytics functions such as listener location.
- Misdirected ad insertion due to misidentification of a listener's location.
- Pre-roll quotas—partial play causing erroneous reporting.

16 TIMELINE OF STREAMING AUDIO AND CODEC DEVELOPMENT

RealAudio (1995): The first popular audio streaming technology was limited by dial-up speeds and an immature codec that produced obvious and objectionable audible artifacts (warbling and “musical noise”) at bitrates supported by dial-up connections. While RealAudio was capable of acceptable audio quality at high bitrates, internet connectivity in 1995 could not support these.

The first version of RealAudio used a proprietary protocol called PNA or PNM to send streaming audio data. RealNetworks later switched to the IETF standardized Real Time Streaming Protocol (RTSP) but they use RTSP only to manage the connection. The actual audio data is sent with their own proprietary RDT protocol, which they initially kept secret. Later, some specifications for the RDT protocol were made public through the Helix Community project.

MP3 (MPEG-2 Audio Layer III; 1993): By 1995, software source code for the MP3 codec became available and led to an explosion of popularity for this codec, which was developed by an international consortium of scientists and engineers. It achieved higher quality for a given bitrate compared to earlier codecs, delivering entertainment-quality audio at bitrates as low as 128 kbps.

The reasonable file sizes of MP3-encoded popular songs led to the massive popularity of illegal music file sharing starting in the mid-1990s. Because files could be downloaded slower than realtime, the quality limitations of dial-up streaming did not apply to MP3 file sharing. This severely impacted recorded music companies, which realized that compressed music had changed their business model forever and that legal streaming and file downloads had to be developed to ensure the industry’s survival. Hence, MP3 was an important driver (if indirect) in the development of today’s streaming industry.

MP3 licensing terms originally required per-stream royalty payments, although the governing MP3 patents are now expired. AAC, introduced in 1997, removed per-stream royalty payments, superseded MP3’s coding efficiency and provided audible transparency at higher bitrates. At this writing, however, MP3 is still widely used for streaming, mainly because of its universal compatibility with players, including older hardware players.

The MP3 standard did not include a method for storing file metadata. In 1996 Eric Kemp developed the ID3v1 protocol. This quickly became the de facto standard for providing metadata in MP3s.

AAC (Advanced Audio Codec; 1997): Many of the same organizations that developed MP3 worked on the improvements that resulted in AAC, which was not backward-compatible with MP3. AAC incorporated many algorithmic improvements over MP3, resulting in a coding efficiency improvement of approximately 20%. This allowed entertainment-quality encoding as low as 96 kbps. AAC is also more versatile than MP3, existing in three “default profiles” that allow trading audio quality against computational complexity. It supports surround formats. AAC became the core technology that was extended in HE-AAC and, later, in Extended HE-AAC (MPEG-D USAC).

AAC metadata is discussed in section 13.2.

Meridian Lossless Packing (MLP) (1997): Developed by Meridian Audio LLC for the DVD-Audio disc format, MLP was one of the first lossless audio codecs to be commercialized and is the basis for the Dolby True HD codec. While it had no direct effect on audio streaming, it paved the way for several royalty-free lossless codecs, such as FLAC, that continue to be used for highest-quality streaming.

Windows Media Audio (WMA; 1999): The Windows Media Audio codec and accompanying ecosystem were developed internally at Microsoft, reportedly to compete with MP3. The company touted “zero-cost streaming” (no per-stream royalties), and digital rights management, which was of great concern to music companies. WMA technology is proprietary to Microsoft.

A WMA file is binary file format and is contained in the ASF file format. It specifies the encoding of metadata about the file similar to the ID3 tags used by the MP3 files.

Later, Microsoft licensed SBR technology from Coding Technologies and incorporated it into an upgraded but short-lived codec called Windows Media Audio Pro.

[https://docs.fileformat.com/audio/wma/]

Ogg Vorbis (2000): Vorbis is a free and open-source software project headed by the Xiph.Org Foundation. The project produces an audio coding format and software reference encoder/decoder (codec) for lossy audio compression. Vorbis is most commonly used in conjunction with the Ogg container format and it is therefore often referred to as Ogg Vorbis. [https://en.wikipedia.org/wiki/Vorbis]

Vorbis uses the Vorbis Comment metadata format. It is similar to ID3 and APE tags, specified by Xiph.org initially for use with (Ogg) Vorbis, and since then also used for, notably, FLAC and Opus.

[https://wiki.hydrogenaud.io/index.php?title=Vorbis_Comment#:~:text=Vorbis%20Comment%20is%20a%20metadata,information%20coded%20in%20UTF%2D8]

FLAC (2001): FLAC (Free Lossless Audio Codec) is a royalty-free audio coding format for lossless compression of digital audio, developed by the Xiph.Org Foundation, and is also the name of the free software project producing the FLAC tools, the reference software package that includes a codec implementation. Digital audio compressed by FLAC's algorithm (which works only on fixed-point audio) can typically be reduced to between 50 and 70 percent of its original size, with dynamically compressed material producing lower amounts of data compression. FLAC-encoded audio decompresses to an identical copy of the original audio data. FLAC has support for fast seeking, plus metadata tagging and images. With the increasing availability of affordable high speed internet service, FLAC (along with several other open-source lossless codecs) enabled extremely high-quality streaming with CD quality (linear **PCM** with 16-bit/44.1 kHz sample rate coding) or better, and up to eight-channel surround sound.

Metadata in FLAC precedes the audio. Properties like the sample rate and the number of channels are always contained in the metadata. It may also contain other information such as the album cover. FLAC uses Vorbis comments (which use UTF-8 text encoding) for textual metadata like track title and artist name.

aacPlus™ (2003): Developed by Coding Technologies for encoding at bitrates of 64 kbps and below, aacPlus combined the AAC Low Complexity profile with “Spectral Band Replication,” which improved coding efficiency at high frequencies. It can provide entertainment-quality streaming as low as 24 kbps.

HE-AAC (2003): aacPlus was standardized as MPEG HE-AAC (High Efficiency AAC) and the aacPlus name was deprecated.

HE-AACv2 (2006): HE-AACv2 added Parametric Stereo to HE-AAC. PS is a method of encoding the spatial properties of stereo content much more efficiently than previously technologies, such as Joint Stereo. In essence, it encodes the sum of the stereo channels as ordinary HE-AAC, and also includes a steering signal (a form of metadata) that reconstructs the stereo spatial impression.

MPEG Surround (2007): MPEG Surround is a lossy compression format for surround sound that provides a backwards compatible method for extending mono or stereo audio services to multichannel audio. It is part of MPEG-D, which specifies MPEG Surround’s metadata capabilities. The total bit rates used for the (mono or stereo) core and the MPEG Surround data are typically only slightly higher than the bit rates used for coding of the (mono or stereo) core. MPEG Surround adds a side-information stream to the (mono or stereo) core bit stream, containing spatial image data. Legacy stereo playback systems will ignore this side-information while players supporting MPEG Surround decoding will output the reconstructed multi-channel audio.

Extended HE-AAC (a standardized profile under MPEG-D **USAC**; 2012): USAC provided further coding improvements over HE-AACv2, particularly at 24 kbps and below. It combined an improved version of HE-AACv2 with a highly efficient speech-oriented codec, allowing the codec algorithm to adaptively switch depending on whether the content was pure speech or not. See USAC.

Ogg Opus (2012): Opus is a lossy audio coding format developed by the Xiph.Org Foundation and standardized by the Internet Engineering Task Force in RFC 6716. It is designed to efficiently code speech and general audio in a single format, while remaining low-latency enough for real-time interactive communication and low-complexity enough for low-end embedded processors. Opus replaces both Vorbis and Speex. Commonly referred to as Opus, it uses the Vorbis Comments metadata format.

17 FUTURE ACTIVITIES

NRSC will investigate ways to advocate for inclusion of NRSC recommendations into relevant standards and specifications. One possible task is to reach out to the Internet Engineering Task Force (IETF), Apple, possibly others to discuss incorporation of NRSC recommendations into the HLS specification.

18 GLOSSARY AND COMPENDIUM OF USEFUL AUDIO STREAMING TERMS

Adaptive Bitrate (ABR)—A technology that detects a player device's processing capability and monitors the available link bandwidth between server and player, automatically adjusting the bitrate of the stream in realtime to achieve the highest audio quality that the user's link and player can support at any given moment without dropouts. This technology paired Multi-Bitrate streaming creates the ideal setup to produce an optimal Quality of Experience for all listeners. HLS and MPEG-DASH support ABR; Icecast and SHOUTcast do not.

Adaptive Multi-Rate Wideband (AMR-WB)—A lossy codec optimized for speech coding. The codec is also known as "HD Voice" to imply that it provides "high definition" voice quality for telephony audio, contrasted with standard digital telephony version of an analog "POTS" (Plain Old Telephone Service) voice-grade call. Specifically, the protocol provides a bandwidth of 7000 Hz, whereas the bandwidth of a traditional voice-grade call is 300 - 3400 Hz.

AMR itself was adopted as the standard speech codec by the 3rd Generation Partnership Project (3GPP) in 1999. The codec's adaptive mechanism uses dynamic link adaptation to select one of eight different bit rates based on cellular radio link conditions.

Advanced Audio Codec (AAC)—A lossy codec that uses a psychoacoustic model to reduce bitrate with minimum subjective loss. It is the successor to MP3 and provides higher audio quality for a given bitrate. It is used by the largest online music distributor and is widely used for high quality live streams. It is one of a family of codecs described in the MPEG-D USAC standard. See Section 12.2: USAC/xHE-AAC™.

AES3—A standard, published by the Audio Engineering Society as AES3-2009 (r2019), for the exchange of digital audio signals between professional audio devices. See section **Error! Reference source not found.**.

AES11—A standard published by the Audio Engineering Society as AES11-2009 that provides a systematic approach to the synchronization of digital audio signals. AES11 recommends using an AES3 signal to distribute audio clocks within a facility. In this application, the connection is referred to as a Digital Audio Reference Signal (DARS). Further recommendations are made concerning the accuracy of sample clocks as embodied in the interface signal and the use of this format as a convenient synchronization reference where signals must be rendered co-timed for digital processing. Synchronization is defined, and limits are given that take account of relevant timing uncertainties encountered in an audio studio. [Wikipedia]

AES67/AES70—An open-standard audio transport mechanism published by the Audio Engineering Society (AES) that allows basic audio interconnection between network audio networks. It does not specify metadata; this is being worked on separately AES67 specifies interoperability requirements for Layer 3 connections. These connections use UDP RTP multicast for audio packets and TCP RTSP for stream initialization. AES67 uses PTP v2 clock for synchronization. AES67 also includes requirements for interoperability of AVB (Layer 2) networks, which must be routed using specialized routers and switches. AES67 does not include "discovery," which is the ability of a given network to automatically discover devices connected to it and to configure itself appropriately, or control protocols that allow devices on the network to be remote-controlled. AES70 ("AES standard for audio applications of networks — Open Control Architecture") attempts to fill in this gap by defining a scalable control-protocol architecture for professional media networks. AES70 addresses device control and monitoring only; it does not define standards for streaming media transport. However, the Open Control Architecture (OCA) is intended to cooperate with various media transport architectures.

AES77—The 2023 elevation of Audio Engineering Society Recommendation AESTD1008.1.21-9 (2021), "Recommendations for Loudness of Internet Audio Streaming and On-Demand Distribution," to an AES Standard with minor editorial changes.

Excessive loudness compromises quality, inconsistent loudness annoys listeners. To resolve these issues, AES77 provides recommendations for establishing and implementing an effective distribution loudness for streaming and on-demand audio file playback. Additionally, it discusses use of Loudness and dynamic range control metadata as part of an "evolutionary process."

AES77 is intended for use by distributors of internet audio streams and on-demand audio files and does not provide recommendations for content production. However, knowledge of AES77 is essential for content

creators and producers because it teaches them how the loudness of their productions will be controlled when disseminated to consumers and discourages the use of unnecessary, quality-compromising peak limiting.

AIFF[11]—Apple audio container format based on the IFF file format, usually used for linear PCM audio in various formats, but not limited to that—it may contain compressed audio, sometimes AIFF-C or AIFC. AIFF is the Apple counterpart to Microsoft .WAV format. Native metadata is limited to only ASCII characters. However, custom metadata chunks can be used, including ID3. AIFF has been superseded by CAF.

AIFF-C—Compressed version of AIFF.

American Standard Code for Information Interchange (ASCII)—The most common character encoding format for text data on computer systems and the internet. In standard ASCII-encoded data, there are unique values for 128 alphabetic, numeric or special additional characters and control codes. ASCII encoding is based on character encoding first used for telegraph data and standardized in 1963.

ASCII was the first major encoding standard for data processing. However, most modern systems now use Unicode, known as the Unicode Worldwide Character Standard, a character encoding standard that includes ASCII encodings.

ASCII encoding is considered obsolete, having been replaced by Unicode. Yet, ASCII characters use the same encoding as the first 128 characters of the Unicode Transformation Format 8, so ASCII text is compatible with UTF-8.

The standard ASCII character set is only 7 bits, and characters are represented as 8-bit bytes with the most significant bit set to 0. The extended ASCII character set includes 127 additional 8-bit characters from binary value from 128 through 255. Unlike the standard ASCII characters, there are multiple versions of the extended character encoding. For example, Microsoft's Windows-1252 code page includes character encoding of the Latin alphabet, which is used for English and various Western-European languages. See Section 7.

AMT—See Intel Active Management Technology.

Apple Lossless Audio Codec (ALAC)—A compressed audio format whose decoder can exactly recover the data applied to the encoder.

Application Programming Interface (API)—A predefined method for computer systems software applications to communicate with each other or with the various layers within an operating system. The Windows API is the exposed entry points created by the operating system to allow software written in various software languages to invoke operating system functions. POSIX is a similar API used on Unix and Unix-like operating systems. Web-APIs allow communication between systems connected by the internet. Whereas a User Interface is usually the character or graphical interface for a human user, an Application Programming Interface is utilized by the software itself to call into other software systems.

AQH–See **Average Quarter-Hour**.

Audio Data Interchange Format (ADIF)—AAC audio data was first packaged in a file for the MPEG-2 standard using Audio Data Interchange Format (ADIF), consisting of a single header followed by the raw AAC audio data blocks. However, if the data is to be streamed within an MPEG-2 transport stream, a self-synchronizing format called an **Audio Data Transport Stream (ADTS)** is used, consisting of a series of frames, each frame having a header followed by the AAC audio data. This file and streaming-based format are defined in MPEG-2 Part 7, but are only considered informative by MPEG-4, so an MPEG-4 decoder does not need to support either format. These containers, as well as a raw AAC stream, may bear the .aac file extension.

Audio Data Transport Stream (ADTS)—A container format specified by MPEG-2/4 for audio data, intended to be used for streamed audio, such as for internet radio. It is a format similar to Audio Data

[11] See https://docs.fileformat.com/audio/aiff/

Interchange Format (ADIF), used by MPEG TS or SHOUTcast to stream audio defined in MPEG-2 Part 7, usually AAC. However, an MPEG-4 decoder may or may not support decoding ADTS. See ADIF above.

Audio-only Streaming—Refers to streams conveying only audio content without accompanying video.

Average Time Spent Listening— An analytical term derived by counting the average number of hours for each listening session of one minute or longer within a specified time period**.**

Average Quarter-Hour (AQH)—"Average Quarter-Hour Persons" is an analytical term conveying the average number of people consuming content for at least five minutes in a given 15-minute period. The "Average Quarter-Hour Rating" is the Average Quarter-Hour Persons estimate expressed as a percentage of the population being measured within a specified market and demographic.

Big Endian—See section 7.10.

Box—See **MP4 Box**.

BS.1770 Loudness—An electrical measurement using the ITU-R BS.1770 algorithm. This estimates the perceived loudness of content, referenced to 0 dBFS, in a unit of "LKFS." The acronyms LUFS (Loudness Units with respect to digital Full Scale) and LKFS (K-weighted Loudness with respect to digital Full Scale) are interchangeable and specify identical values. The algorithm consists of a K-weighting filter feeding a level-gated RMS detector followed by a logarithmic converter scaled such that a loudness change of 1 LU corresponds to a level change of 1 dB. The first stage of the filter is a high frequency 4dB shelving boost that accounts for the acoustical effects of the head. The second stage is a low-frequency roll off based on "B-weighting," which roughly approximates the 70 phon equal-loudness contour (see ISO 226:2003). [Credit: from AES TD-1008]

Broadcast WAV File (BWF/BEXT)—An extension of the Microsoft WAV audio format, specified by the European Broadcasting Union (EBU) in EBU-Tech 3285 and by the ITU in Recommendation ITU-R BS.1352-3. The purpose of this file format is the addition of metadata to facilitate the seamless exchange of sound data between different computer platforms and applications. It specifies the format of metadata, allowing audio processing elements to identify themselves, document their activities, and it supports timecode to enable synchronization with other recordings. The BWF metadata is stored as extension chunks. These can include the original Bext chunk (Broadcast Extension - 'bext', which includes loudness metadata), an iXML chunk ('iXML'), a quality chunk ('qlty'), an MPEG audio extension chunk ('mext'), a Peak Envelope chunk ('levl'), a link chunk ('link'), an axml chunk ('axml'), and LIST INFO, which supports ASCII only. There is no UTF-8 support, making this a poor choice for modern archiving. However, BWF is still used for most legacy archiving such as the Library of Congress. While loudness metadata is always part of the file, practitioners should note that it may not have been set.

Byte Order Mark (BOM)—Special Unicode characters to signify several parameters, such as byte order, or endianness of a Unicode text file.

Cart—Short for "cartridge," referring to NAB endless-loop analog audio tape cartridges used for many years in broadcast audio studios for commercials and music prior to adoption of current digital playout systems. Many playout system software vendors have used the term out of legacy familiarity for software elements that have similar functionality.

Cascading Style Sheet (CSS)—A style sheet language used to describe the presentation of a document written in a markup language such as HTML or XML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of content and presentation, including layout, colors, and fonts, improving content accessibility and providing more flexibility and control in the presentation characteristics. It can be used to style and control web players. https://en.wikipedia.org/wiki/CSS

CD Digital Audio Extraction (CD DAE)—The process of extracting raw digital audio from a Red Book Audio CD into an audio file, also known as CD Ripping. It is best to extract to a linear PCM or lossless format for accuracy. It is not a good idea to save as MP3, as data loss occurs, resulting in poor audio quality. There are many CD DAE software applications available with various features. Few get accuracy and metadata correct, so this software should be vetted very carefully, as this is the starting point and a major determining factor in the overall quality of audio and metadata presentation of content. See section 13.8.

CENC—See Common Encryption.

Character Encoding—The procedure of assigning a digital representation to characters, typically used in human language, for the purpose of digital transmission and interchange between computer systems. These digital representations typically utilize an agreed-upon standard. For example, ASCII, UTF-8, UTF-mb4 are character encoding standards used for the digital transmission of text data. Section 7 treats this is detail. See also **ASCII** and **Code Page** for additional information.

Character Map—The process of converting characters between encodings.

Chunk—A block of audio or metadata within an audio file or stream. Commonly referred to as a block, frame or segment. A chunk may contain headers that provide specific information about its contents.

Code Page—In computing, a code page is a character encoding and as such it is a specific association of a set of printable characters and control characters with unique numbers. Typically each number represents the binary value in a single byte. IBM introduced the concept of systematically assigning a small, but globally unique, 16-bit number to each character encoding that a computer system or collection of computer systems might encounter.

Unicode is an effort to include all characters from all currently and historically used human languages into single character enumeration (effectively one large single code page), removing the need to distinguish between different code pages when handling digitally stored text. Unicode tries to retain backwards compatibility with many legacy code pages, copying some code pages in the design process. An explicit design goal of Unicode was to allow interchange between all common legacy code pages, although this goal has not always been achieved. Some vendors, namely IBM and Microsoft, have anachronistically assigned code page numbers to Unicode encodings. This convention allows code page numbers to be used as metadata to identify the correct decoding algorithm when encountering binary stored data. Modern operating systems are fully Unicode-compliant, rendering code pages unnecessary. [Wikipedia except for the last sentence]

When you open a command box in Windows, Unicode is not enabled by default and must be invoked by a command that specifies the Unicode code page.

Codec—An acronym created by eliding COder and DECoder. A codec reduces the number of bits needed to represent content for storage and/or transmission compared to straightforward PCM (pulse code modulation) such as that used for CDs and AES3 digital audio distribution within a facility. Codecs can be lossy or lossless. Lossless codecs' outputs are identical to their inputs. They are like zip files, discarding mathematically redundant parts of the content. Lossless codecs can rarely compress the content by more than 2:1, and content with limited dynamic range further limits compression effectiveness. Lossy codecs achieve far greater compression than lossless codecs. They maximize perceptual quality at a given output bitrate by discarding what their designers consider to be the least perceptually important information. Most lossy codecs (such as AAC) identify and discard this information via a psychoacoustic model that mimics human auditory perception and then use mathematical methods to further reduce redundancies. Some lossy codecs can be perceptually transparent as a given bitrate, meaning that human listeners cannot statistically distinguish their input from their output when tested according to ITU-R BS.1116-1: *Methods for the Subjective Assessment of Small Impairments in Audio Systems including Multichannel Sound Systems*.

Code Point—Code points allow abstraction from the term "character" and are the atomic unit of storage of information in a text encoding. Most code points represent a single character, but some represent information such as formatting and control.

Common Encryption (CENC)—The Common Encryption Scheme (CENC) specifies standard encryption and key mapping methods that can be utilized by one or more Digital Rights Management and key management systems (DRM systems) to enable decryption of the same file using different DRM systems. The scheme operates by defining a common format for the encryption related metadata necessary to decrypt the protected streams, yet leaves the details of rights mappings, key acquisition and storage, DRM compliance rules, etc. up to the DRM system or systems supporting CENC. The ISO/IEC 23001-7 standard defines four Common Encryption modes. [https://docs.unified-streaming.com/documentation/drm/common-encryption.html]

Compact Disc Rip Metadata—CD rip metadata can be extracted from CD text if it is present on the CD being ripped. CD-Text is an extension of the Red Book Compact Disc specifications standard for audio CDs. It allows storage of additional information (e.g., album name, song name, and artist name) on a standards-compliant audio CD. The specification for CD-Text was included in the Multi-Media Commands Set 3 R01 (MMC-3) standard, released in September 1996. It was also added to new revisions of the Red Book (CD standard). The actual text is stored in a format compatible with Interactive Text Transmission System (ITTS), defined in the IEC 61866 standard. [Wikipedia]

Common Media Application Format (CMAF)—A standardized format to simplify the delivery of HTTP-based streaming media, reducing cost and complexity. CMAF can be used in DASH or HLS. It is standardized in ISO/IEC 23000-19:2020 [Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media].

Content Distribution Network (CDN)—A resource to provide streaming and on-demand services for content providers that choose not to do so on their own. Many CDNs are feature-limited, so they should be chosen carefully.

Container—A data structure that encapsulates media content and (optionally) metadata in a file format that a compatible player can read to extract the content. Examples include RIFF/WAV, AIFF, and MP4. In the context of streaming, this term can also refer to a streaming transport protocol such as HLS or ICY.

Content Management System (CMS)—Software that helps developers create, manage, and modify content on a website without the need for specialized technical knowledge. Instead of developers' building their own systems for creating web pages, storing images, and other functions, the content management system handles all of those basic infrastructure tasks. [https://kinsta.com/knowledgebase/content-management-system/]

Core Audio Format (CAF)—A container for storing audio, developed by Apple Inc. It is compatible with Mac OS X 10.4 and higher. Core Audio Format is designed to overcome limitations of older digital audio formats, including AIFF and WAV. A .caf container can contain many different audio formats, metadata tracks, and much more data. It is not limited to a 4 GB file size and uses 64-bit file offsets. The native metadata is UTF-8. Custom metadata chunks can be used, including ID3

CR/LF (Carriage Return Line Feed)—Terminal (TTY) control characters used in text data strings to end a line of text and add an additional line immediately below the current line. Different operating systems use different syntax.

CSS—see **Cascading Style Sheet**.

Cume Persons—The total number of unique listeners who connect to content during the course of a daypart for at least five minutes. Depending on the analytics technology employed, this number may be an estimate based on a sample or it may actually count each unique listener.

Cume Rating—The Cume Persons audience expressed as a percentage of all listeners estimated to be in the specified demographic group. "Exclusive Cume" is the number of unique listeners who listen to only one stream or station during the daypart reported.

DASH—See Dynamic Adaptive Streaming over HTTP.

DASHdirect™ (StreamS/Modulation Index, LLC)—A fully CMAF-compliant DASH streaming encoder implementation. The encoder output is fully segmented DASH with metadata. The encoder connects to various servers for delivery to DASH player clients.

DAV, DAVS—See Distributed Authoring and Versioning and WebDAV.

Decoder—In streaming, a component that is part of the player client, used to reassemble the media content for playout. The term is also used to refer to the decoder element in a codec such as AAC. See **codec**.

Designated Market Area (DMA)®—In audience analytics, a Nielsen-trademarked term. The DMA is composed of sampling units (counties or geographically-split counties) and is defined and updated annually by Nielsen.

dialnorm—A Dolby codec metadata parameter that indicates the loudness of the audio and is used in the normalization of the audio after decoding enabling consistent and predictable loudness across sources and content. It was the first implementation of “Loudness” metadata, now commonly supported in the streaming and broadcast ecosystem and discussed in Loudness and dynamic range control metadata. For audio containing speech or dialogue, the dialnorm is typically set to the average dialogue level (hence the name dialnorm), whereas for music it is set to the average loudness based on Recommendation ITU-R BS.1770-4. It has a range of -1 to -31 LKFS and provides the amount of gain reduction that should be applied to normalize the audio. See ATSC A52(2018) for AC-3/E-AC-3 and ETSI TS 103 190-1 (2018-02) for AC-4. While these codecs have previously been used almost exclusively for sound-with-picture, they are now also used for immersive music and podcasts.

Digital Rights Management (DRM)—The management of legal access to digital content to allow it to be protected from piracy and monetized by rights holders. Generically, DRM encrypts content and may also employ metadata. Common DRM systems used in audio streaming include FairPlay (Apple), PlayReady (Microsoft), and Widevine (Google).

Distributed Authoring and Versioning (DAV, DAVS, WebDAV)—An extension of HTTP/1.1 described in RFC 2518 and RFC 3253 (which added versioning control to RC 2518). WebDAV can be considered a protocol. It contains a set of concepts and accompanying extension methods to allow to allow reading, writing, creation of directories, and file and directory deletion on a web server file system across the HTTP/1.1 protocol. Instead of using NFS or SMB, WebDAV offers file transfers via HTTP. The basic idea is that a WebDAV-compliant web server can act like a generic file server; clients can “mount” shared folders over HTTP that behave much like other network filesystems (such as NFS or SMB).

This is the preferred method for streaming direct HLS/DASH to a web server. It does not require the complications of deploying and managing an additional FTP server; FTP traffic has become difficult over the public internet because there is additional Stateful Packet Inspection involved for security reasons, increasing latency.

Dynamic Adaptive Streaming over HTTP (DASH) (MPEG-DASH)—A high performance segmented streaming transport protocol used for live and file (on-demand) streams. Like HLS, it splits streams into smaller segments and optionally encodes segments at different quality levels for different network conditions to increase reliability and decrease distribution cost. It is standardized in [ISO/IEC 23009-1:2022].

Dynamic Ad Insertion (DAI)—Targeted ad insertion or replacement that is unique to a given listener and usually inserted by the CDN or aggregator. This can be based on listener location, listener tracking, browsing preferences, etc. See section 11.

Dynamic Range Control (DRC)—The process of continually adjusting audio signal level to control the loudness difference between loud and soft passages. This can help overcome noise in the listening environment, meet the dynamic range capability of the playback equipment, or both. The term “dynamic range control” typically implies that the codec’s encoder generates the gain control signal as metadata and that the player device’s decoder acts on this metadata. While this can also be described as “dynamic range compression,” that term is more commonly used when a single device (like the player) generates the gain control signal from an internal algorithm applied to the incoming audio and does not exploit DRC metadata. See Loudness and dynamic range control metadata .[from AES TD1008]

ECMAScript—A general-purpose programming language, best known as the language embedded in web browsers. However, it has also been widely adopted for server and embedded applications. ECMAScript is based on several originating technologies, the most well-known being JavaScript and JScript. As of this writing it is in its 13th edition and is standardized in ISO/IEC 16262.

Elementary Stream (ES)—An elementary stream, as defined by the MPEG communication protocol, is usually the output of an audio encoder or video encoder. An ES contains only one kind of data (e.g., audio, video, or closed caption). An elementary stream is often referred to as "elementary," "data," "audio," or "video" bitstreams or streams. The format of the elementary stream depends upon the codec or data carried in the stream but will often carry a common header when packetized into a packetized elementary stream. [Wikipedia] In a fully-implemented HLS ES, ID3v2.4 metadata frames are included.

Encoder—A stream's transmitter, which formats the audio and metadata inputs so they can be streamed via a given protocol or container, and which may include a codec. The codec ingests media content and decreases its bitrate for transmission. Codecs can be lossless, such as FLAC, or lossy, such as MP3 or AAC. PCM is special case because no compression is used, so the stream merely passes the input bitstream within the stream's container. See **codec**.

Encoding Format—A name for the codec technology used to encode and decode audio (e.g., AAC, MP3, FLAC). The decoder is typically standardized, while the encoder may be improved over time to increase coding efficiency. To allow playback, the player application automatically recognizes the encoding format.

Encrypted Media Extensions (EME)—This extends HTMLMediaElement [HTML51], providing APIs to control playback of protected, encrypted content. License/key exchange is controlled by the application, facilitating the development of robust playback applications supporting a range of content decryption and protection technologies. This specification does not define a content protection or Digital Rights Management system. Rather, it defines a common API that may be used to discover, select and interact with such systems as well as with simpler content encryption systems. Implementation of Digital Rights Management is not required for compliance with this specification: only the Clear Key system is required to be implemented as a common baseline. The API supports use cases ranging from simple clear key decryption to high value video (given an appropriate user agent implementation). License/key exchange is controlled by the application, facilitating the development of robust playback applications supporting a range of content decryption and protection technologies. [https://www.w3.org/TR/encrypted-media/]

Encryption—A method of obfuscating content in a stream or file to enforce security, so that a player that does not have access to complementary decryption protocol and decryption keys cannot play the content. Examples include FairPlay/PlayReady/Widevine.

Endianess—See section 7.10.

Event Message (emsg)—CMAF media may contain in-band media timed events in the form of Event Message (emsg) boxes in ISO BMFF files. fMP4 segmented streaming uses emsg ID3v2.4 for synchronous metadata. This is the preferred method for sending Now-Playing and control metadata. emsg is specified in MPEG-DASH (ISO/IEC 23009-1:2022).

Extensible Markup Language (XML)—A markup language and file format for storing, transmitting, and reconstructing arbitrary data. It defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications, all of them free open standards, define XML [Wikipedia].

Filename Metadata—Metadata generated from the filename of the associated audio file. It is subject to the limitations discussed in section 6.3: Certain characters are reserved for operating systems and cannot be used for metadata. Furthermore, the traditional artist-title separation character, the hyphen, will cause trouble if either artist and/or title contain a hyphen.

File Transfer Protocol (FTP/FTPS)—A standard communication protocol used for the transfer of computer files from a server to a client on a computer network. FTP is built on a client–server model architecture using separate control and data connections between the client and the server. FTP users may authenticate themselves with a clear-text sign-in protocol, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it. For secure transmission that protects the username and password, and encrypts the content, FTP is often secured with SSL/TLS (FTPS) or replaced with SSH File Transfer Protocol (SFTP). (Wikipedia)

Flash—Adobe proprietary technology, all versions of which are now deprecated due to security vulnerabilities. No current browser supports it.

FLV—Flash Video Format, a container file format used to deliver digital video and/or audio content over the internet to Adobe Flash Player. Now deprecated and unsupported because of security issues.

Free Lossless Audio Codec (FLAC)—A royalty-free compressed audio codec whose decoder can exactly recover the data applied to the encoder.

Global Positioning System (GPS)—A satellite-based radionavigation system owned by the United States government. It is one of the global navigation satellite systems (GNSS) that provides geolocation and time

information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. The GPS satellites carry very stable atomic clocks that are synchronized with one another and with the reference atomic clocks at the ground control stations. This allows GPS to be used as a very precise time and frequency standard, which is useful in synchronizing devices on local area networks and the internet. One important application is disciplining a Grandmaster Clock in a network that uses the Precision Time Protocol. In practical use, GPS-derived time is typically accurate to about 160ns and relative frequency deviation can be less than 10^{-12} when averaged over several hours. In the short-term, GPS time and frequency reception is noisy, so highly accurate frequency standards often use a GPS-disciplined quartz crystal or atomic clock to exploit the low phase noise and short-term frequency drift of these technologies while allowing GPS to correct long-term drift.

GPAC—An open-source multimedia framework developed for research and academic purposes, used in many media production chains. The project covers various aspects of multimedia, from multimedia packaging and distribution using formats such as MP4 or MPEG-2 TS to interactive presentation technologies (graphics, animation and interactivity). It provides three sets of tools based on a core library called libgpac: (1) A multimedia packager, called MP4Box, (2) a generic media pipeline orchestrator, called gpac, used to build complex media processing sessions (players, transcoders, streamers, packagers, servers, etc.), and (3) bindings for Python and NodeJS. [https://gpac.wp.imt.fr/home/]

Grandmaster Clock—The reference PTP (IEEE 1588) generator (media clock) in an audio-over-IP network that supports PTP. A grandmaster clock receives time information from an external time reference, most commonly a global navigation satellite system source, and distributes it downstream to other clocks on the network. The grandmaster clock should be the source of wordclock and other hardware sample frequency references within a given facility.

HD Radio™—Trademark of Xperi, Inc. (formerly iBiquity Digital Corporation) for the in-band on-channel (IBOC) digital AM and digital FM transmission technology authorized by the FCC. The use of this term in NRSC documents shall be interpreted as the generic term “IBOC” and shall not be construed as a requirement to adhere to undisclosed private specifications that are required to license the HD Radio name from its owner.

HE-AAC / HE-AAC v2 / aacPlus™(Dolby/Coding Technologies)—"High-Efficiency AAC" is a lossy audio codec for low bitrate streaming, expanding on the quality vs. bitrate gains made by standard AAC by adding Spectral Band Replication (v1 and v2) and Parametric Stereo (v2) technologies to a core AAC codec. It provides good subjective quality at bitrates as low as 32 kbps and is commonly used.

HEX—Data consisting of numerals in a hexadecimal format (radix/base 16). A given numeral may be represented by the digits 0 through 9 or by alphabetic characters A, B, C, D, E, and F. In many cases, multiple HEX numerals represent characters. Modern character encoding uses the UTF-8 protocol, defined in the Unicode Standard.

Hex Dump—A display or printout that displays each character in a file or data stream in HEX format. In a hexadecimal view of data, a two-digit number represents each byte (8 bits). Hex dumps are commonly organized into rows of 8 or 16 bytes, sometimes separated by whitespaces. Some may show the hexadecimal memory address at the beginning. Hex dumps are useful to verify that data commands are executing properly and on time. There are examples in this document.

Hidden Metadata-Insertion Cue-Control—See Metadata for supplemental control and information.

HLS—See **HTTP(S) Live Streaming**.

HLS Codecs—Supported Packed Audio formats in legacy HLS Elementary Stream (ES) are AAC with ADTS framing [ISO_13818_7], MP3 [ISO_13818_3], AC-3 [AC_3], and Enhanced AC-3 [AC_3]. Modern fMP4 HLS-supported audio codecs are MPEG4 AAC, USAC/Extended HE-AAC, xHE-AAC™, ALAC Lossless, FLAC Lossless, and Dolby Atmos. Additional codecs can be added to fMP4 HLS because standards-based ISO/MPEG-4 containers are used.

HLSdirect™ (StreamS/Modulation Index, LLC)—A fully CMAF-compliant HLS streaming encoder implementation. The encoder output is fully segmented HLS with metadata and connects to various servers for delivery to HLS player clients.

HTML (Hypertext Markup Language)—The standardized markup language used to specify all elements in a document intended for display in a web browser. The current version is HTML5.

HTML5—The fifth major HTML version that is a World Wide Web Consortium (W3C) recommendation. This version is more compatible for multimedia. The current specification is known as the HTML Living Standard. It is maintained by the Web Hypertext Application Technology Working Group (WHATWG), a consortium of the major browser vendors

HTML5 Audio Tag—The HTML5 audio element <audio> is used for basic file playback from within an HTML document. Although some browsers will play an ICY stream using this method, there is no metadata available and it is subject to the unreliability of a constant stream. Many naïve developers choose this method because of simplicity. However, metadata must then be pushed out-of-band, making timing unreliable. Moreover, some browsers fill their cache with audio, so long listening times cause problems with memory management, making the <audio> tag unsuitable for live ICY streams.

To overcome these obstacles, the Web Audio API, with MSE and segmented streaming should be used. Note that the HTML <audio> tag does not support segmented media by itself. but **WebAudio** and MSE can be pushed into it.

HTML5 Encrypted Media Extensions (EME)—A W3C specification for providing a communication channel between web browsers and the Content Decryption Module (CDM) software that implements digital rights management (DRM). EME is based on the HTML5 Media Source Extensions (MSE) specification. [Wikipedia]

HTML5 Media Source Extensions (MSE)—This specification extends HTMLMediaElement to allow JavaScript to generate media streams for playback. Allowing JavaScript to generate streams facilitates a variety of use cases like adaptive streaming, time-shifting live streams, and real-time metadata display. [https://www.w3.org/TR/media-source-2/].

HTTP—See HyperText Transfer Protocol**.**

HTTP(S) Live Streaming (HLS)—An HTTP-based segmented, adaptive-bitrate streaming protocol developed by Apple Inc. and released in 2009. Support for the protocol is widespread in media players, web browsers, mobile devices and streaming media servers. HLS provides high-performance and reliability, and is used for live and file streams. Similar to DASH, it also splits streams into smaller segments and optionally encodes segments at different quality levels for different network conditions to improve reliability at lower distribution cost. It supports in-band, synchronous, extensible metadata, required for on-time now-playing PAD and content insertion metadata. HLS has been IETF, RFC, and ISO standardized.

HyperText Transfer Protocol (HTTP)—The communications protocol used to make connections between browsers and Web servers on the internet or on a local network (intranet). The primary function of HTTP is to establish a connection with the server and send HTML pages back to the user's browser. It is also used to download data from the server either to the browser or to any requesting application that uses HTTP. More specifically, HTTP is the set of rules for transferring files—such as text, images, sound, video and other multimedia files—over a network. HTTP is an application protocol that runs on top of the TCP/IP suite of protocols, which forms the foundation of the internet. HTTPS adds end-to-end encryption to HTTP.

iAMT—A realization of Intel Active Management Technology that may be included for free in devices; however, the full capabilities of iAMT, including encrypted remote access via a public key certificate and automatic remote device provisioning of unconfigured iAMT clients, are not accessible for free for iAMT-equipped devices.

IBOC—See **In-Band On-Channel**.

Icecast—An open-source streaming media project released as free software and maintained by the Xiph.org Foundation using a modified ICY protocol. It is similar to SHOUTcast, but different enough to require player clients to specifically support it. It was developed as an open-source alternative to proprietary SHOUTcast. It is basically a hack of the HTTP protocol and was never formally standardized. It is widely used, but its many different implementations require player clients to support the specific implementation. It does not support synchronous metadata, precluding creation of high-performance web player clients.

ICY (I Can Yell)—A streaming audio protocol that has gained popularity due to its simplicity. The ICY protocol is a superset of the HTTP protocol and is designed for media streaming. It is identified by headers in the HTTP packet that starts with "icy-".

icy-metaint—The number of bytes of media stream data between each metadata chunk in an ICY stream. This is ICY asynchronous metadata and is server-definable.

ID3—A container format for metadata. The format is commonly called a “Tag.” The data, which is extensible, usually has “now playing information.” Because it is extensible, it can have many other fields a content provider would want, including playout cues.

ID3 v2.4 is the currently accepted format, supports UTF-8 and is used in HLS/DASH streaming. The ID3 frames *must* be parsed with an ID3 binary parser that supports UTF-8mb4 for completeness. Naïve ASCII parsers will produce garbage characters when parsing extended, international, or emoticon characters.

ID3v2.3 is the ID3 tag most commonly used for MP3 files. It is UTF-16 but is not compatible with ID3v2.4.

Impression—see Page Impression

IMSC1—See Internet Media Subtitles and Captions.

In-Band—In streaming, a method that accommodates two or more sets of data sent in the same transport and/or container. It usually refers to audio and associated PAD/metadata. In radio, digital data transmitted in the same licensed RF channel as an accompanying analog transmission. See IBOC.

In-Band On-Channel (IBOC)—An RF transmission method of broadcasting analog and digital radio signals simultaneously on the same FM or AM frequencies. This methodology is widely used in North America for reasons that include advantageous spacing of channels as well as relative ease of implementation and introduction to the general public. Digital radio standards allow for rich metadata and graphics (Program-Associated Data) that analog transmission and receivers do not support using the associated Radio Data System (RDS). HD Radio™ is an example of IBOC.

Ingest—The process of collecting content and (optionally) metadata from a source and formatting the collected data to its intended purpose, which may be for immediate use or storage in a database. Data can be received in realtime or ingested in batches. An example is extracting the audio data and CD text from an audio CD and transferring it to a playout system that uses hard drives or cloud storage. In streaming, the term can apply to the server-to-encoder connection.

Injected content—Content from a secondary source, such as targeted ad insertion, that replaces content from a primary source from time to time.

Intel Active Management Technology (AMT)—Hardware and firmware for remote out-of-band management of Intel computers. It runs on the Intel® Management Engine, a microprocessor subsystem not exposed to the user and intended for monitoring, maintenance, updating, and repairing systems. Out-of-band (OOB) or hardware-based management is different from software-based (or in-band) management and software management agents. AMT is Intel’s method for supplying functionality similar to IPMI.

Intelligent Platform Management Interface (IPMI)—Defines a set of interfaces for out-of-band management of computer systems, control, and monitoring of their operation, independent of the host system CPU, firmware, and operating system. This facilitates complete remote administration of computer systems, including power control and reboot.

Interactive Advertising Bureau (IAB)—An interactive advertising association. Its activities include evaluating and recommending standards and practices, fielding research to document the effectiveness of the interactive medium and educating the advertising industry about the use of interactive advertising. Membership includes companies that are actively engaged in, and support the sale of interactive advertising. For their product to be tracked and monetized, broadcasters and netcasters should comply with IAB standards. [https://www.iab.com/news/interactive-advertising-bureau-iab/]

International Atomic Time (TAI)—A weighted average of the indications of around 450 atomic clocks in approximately 80 national timing laboratories around the world. The relative stability of TAI is around one part in 10^{16}.

International Standard Recording Code (ISRC)—The international identification system for sound recordings and music video recordings. Encoded ISRCs provide the means to automatically identify recordings for royalty payments. ISRCs are widely used in digital commerce by download sites and collecting societies. Each ISRC is a unique and permanent identifier for a specific released track, independent of the format on which it appears (CD, audio file, etc.) or the rights holders involved. For example, released versions of an original studio recording such as the original mastering, a remastered version, a remix, etc. will all receive different ISRCs if they are assigned correctly. An ISRC can never represent more than one unique recording. As a matter of good practice, only one ISRC should be issued to a track, but this practice is frequently violated. An ISRC can also be permanently encoded into a product as its digital fingerprint. The Recording Industry Association of America has been appointed by the International ISRC Agency to oversee the ISRC system within the United States and its territories. Further information on the ISRC can be found at https://usisrc.org/about/index.html.

Internet Engineering Task Force (IETF)—The leading internet standards body. It develops open standards through open processes. A large open international community of network designers, operators, vendors, and researchers, the IETF focuses on the evolution of the internet architecture and the smooth operation of the internet. The Internet Architecture Board (IAB) and the Internet Research Task Force (IRTF) complement the work of the IETF by, respectively, providing long-range technical direction for internet development and promoting research important to the internet's evolution. [https://www.internetsociety.org/about-the-ietf/]

Internet Media Subtitles and Captions (IMSC1)— A file format for representing subtitles and captions. It uses XML to describe content, timing, layout, and styling. IMSC is very similar to HTML and CSS in concept. It is standardized by the W3C as "TTML Profiles for Internet Media Subtitles and Captions 1.0.1," and used around the world by content producers, online services, and traditional broadcasters. Each IMSC document is self-contained and combines content, timing, layout and styling information. The content of the document is structured using tags similar to those used in HTML. [https://developer.mozilla.org/en-US/docs/Related/IMSC]

Interstitial—One or more short, recorded elements used in stream production (i.e., bumpers, jingles, promos, etc.). Interstitials are typically 60 seconds or less in length.

ISMC—A Smooth Streaming Client Manifest File used by Smooth Streaming, an extension for Microsoft's IIS Web server for streaming multimedia content; hosted on the IIS Web server. The ISMC file contains information about the codecs, resolutions, and fragments (the video chunks in .ISMV files) for hosted multimedia files, saved in an XML format.

ISO BMFF—ISO Base Media File Format, based on the original Apple .MOV file format. It is standardized in ISO/IEC 14496-12. The ISO BMFF is the basis for the MP4 family of file formats.

ISO Box—See MP4 Box.

iTunes Music Store (iTMS) format—A metadata format developed by Apple Computer. It is commonly used in MP4/M4A containers, which are custom ISO boxes, not ID3v2. See AAC. (Note that the iTunes Music Store is currently known as the Apple Music Store.)

Java—A high-level, object-oriented, statically-typed programming language that is platform-independent and designed to be robust and secure. Java supports multithreading, automatic memory management through garbage collection, exception handling, and has a vast standard library that provides developers with a wide range of pre-built functions and classes for common programming tasks. Java programs are compiled into bytecode, that can be executed on any Java Virtual Machine (JVM), making it possible to run Java code on any platform without the need for recompilation. Additionally, Java supports multi-threading and exception handling, making it suitable for developing complex and reliable applications, including enterprise-grade systems, web applications, and mobile apps.

JavaScript—A high-level, dynamically-typed, interpreted programming language that is primarily used for creating interactive web pages and web applications. It is a multi-paradigm language that supports both object-oriented and functional programming styles. JavaScript is executed by web browsers and provides powerful client-side scripting capabilities, allowing developers to manipulate web page content and respond to user events. JavaScript can also be used on the server-side through the use of frameworks such as

Node.js. The language features a vast array of built-in functions and a large ecosystem of libraries and frameworks that make it easy to develop complex web applications. It includes powerful audio APIs called WebAudio and WebRTC. Additionally, JavaScript is platform-independent and runs on all major web browsers, making it a popular choice for front-end web development. JavaScript was based on the ECMAscript standard [Standard #].

JPEG (aka JPG)—A commonly used method of lossy compression for digital images. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality. JPEG typically achieves 10:1 compression with little perceptible loss in image quality. The term "JPEG" is an acronym for the Joint Photographic Experts Group, which created the standard in 1992. JPEG was largely responsible for the proliferation of digital images and digital photos across the internet and later social media.

JavaScript Object Notation (JSON)—A lightweight data-interchange format. JSON is based on a subset of the JavaScript programming language and is often used for transmitting data between a client and a server in web applications. JSON consists of key-value pairs, where the keys are strings and the values can be a variety of data types, including strings, numbers, booleans, arrays, and other objects. JSON is widely supported by programming languages and platforms, making it a popular choice for data exchange in web development. It is easy for humans to read and write and easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON can be an alternative to Extensible Markup Language (XML) for PAD/metadata. Under certain circumstances, it produces slightly lighter-weight data than XML.

Latency—In the context of streaming, the time delay between when data (metadata or content) is applied to the streaming encoder and when the player presents it. Without synchronous metadata the latency of the metadata and audio can be different, causing metadata and audio to be out of sync at the player. In the context of codecs, latency is the time delay between when audio is applied to the codec and when it is decoded. Each codec algorithm has an intrinsic latency. Some codecs allow users to change the latency to trade off robustness against low delay.

Little Endian—See section 7.10.

LKFS—Stands for “Loudness, K-weighted, relative to Full Scale.” A standard loudness unit of measurement. Sometimes referred as LUFS, which is identical. Defined in ITU-R BS.1770 and used in EBU R-128, AES TD1008, and ATSC A/85, among others. See BS.1770.

Log— In the context of this document and when used as a noun, it typically denotes a record of specified actions or events. As a verb (“to log”), it denotes making a log. Because “log” has different meanings in different industry segments, it can be a source of considerable confusion.

Broadcasters use several types of logs. Examples are program, music, and traffic or commercial logs. Generally, the music log is merged with the traffic log to produce a program log. In playout systems, these logs are called Playlists. This Playlist is not to be confused with an HLS Playlist, which is produced by a true HLS streaming media encoder and describes the streaming segment order.

In broadcasting and netcasting, metadata allows a log to be generated to create royalty and affidavit reports.

In the context of electronic hardware and software, logs can refer to records of events such as actions, network connections, and errors. In Windows computers, such logs can be examined in Event Viewer.

Low Overhead Audio Transport Multiplex (LATM)—See LOAS.

Low Overhead Audio Stream (LOAS)—A self-synchronizing format defined in MPEG-4 Part 3 that encapsulates not only AAC, but any MPEG-4 audio compression scheme such as Unified Speech and Audio Coding. This format was defined for use in DVB transport streams when encoders use either SBR or parametric stereo AAC extensions. However, it is restricted to only a single non-multiplexed AAC stream. This format is also referred to as a Low Overhead Audio Transport Multiplex (LATM), which is just an interleaved multiple stream version of a LOAS. LOAS and LATM are standardized in ISO/IEC 14496-3.

Lossless Stream—A high-bitrate uncompressed stream or a losslessly-compressed stream that uses a codec (like FLAC) whose decoder can exactly recover the data applied to the encoder.

Lossy Stream—A stream using a lossy codec such as MP3 or AAC. The codec attempts to minimize the loss in audio quality perceived by the ear while decreasing bitrates below those needed by lossless codecs. The codec typically incorporates a psychoacoustic model to achieve this goal.

LUFS—"Loudness Units Relative to Full Scale," synonymous with **LKFS**.

M4A—ISO Standard MPEG-4 Media File Container format that specifies audio only (See MP4). Metadata is usually carried in the ITMS (iTunes Music Store) format, not id3. Although id3 can be used in a separate extensible ISO Box, it is not widely supported across various player applications.

M4V—A derivative of the MP4 container format developed by Apple that designates video content.

M4S -fMP4—Fragmented—An ISO Standard MPEG-4 Media File Container format that specifies MP4 Segments. It must be used with an MP4 initialization file. M4S is the segment file format used for CMAF HLS/DASH and is referred to as ISO BMFF. Metadata is carried in an extensible emsg ISO Box as ID3v2.4 (UTF-8mb4).

M4P—An MPEG4 DRM-Protected File using the AAC codec. M4P files use a proprietary DRM technology created by Apple Computer, originally for use with iTunes purchased files although no longer used there. Apple Music uses M4P.

If file conversion is necessary, it is best to convert to a PCM format or to another DRM-free AAC-LC, *not* MP3.

Metadata—Data about the data: Metadata is data that accompanies the media content and describes aspects of the content. Metadata can hold the “Now-Playing” information, commercial cues, time code and any other information needed to play the stream. The two basic methods of conveying metadata are file-based and data-based. Both methods may use templates to define the data structure, facilitating software interoperability (i.e., playout to encoder). File-based metadata is generally associated with a file transfer fetched with a defined query time by the player. Data-based metadata is pushed in-band by the encoder to the player; if the implementation is correct, the metadata will be synchronous.

Metaint—See **icy-metaint**.

Manifest—MPD—DASH—The MPEG-DASH manifest describes what type of content is being streamed and how it should be played back by the client software. The manifest file contains information about each segment, such as how long it is and its format. It also contains a list of all the segments that make up the entire video or audio stream. [https://www.keycdn.com/]

Media Source Extensions (MSE)—A W3C specification that allows JavaScript to send byte streams to media codecs within Web browsers that support HTML5 video and audio. It is compatible with, but should not be confused with, the Encrypted Media Extensions (EME) specification. Neither requires the use of the other, although many EME implementations are only capable of decrypting media data provided via MSE.

Metro—In audience analytics, Metro specifies a city (or cities) whose population demographic is specified as that of the central city together with the county (or counties) in which it is located. The Metro also includes contiguous or additional counties when the economic and social relationships between the central and additional counties meet specific criteria.

Middleware—A software application or script that provides a translation layer between other software applications. It is commonly used to connect playout system software metadata output to a destination such as a streaming encoder.

Mid-Roll—An online advertising term describing insertion of interstitial content in the middle of a program. See also Pre-roll. [https://sproutsocial.com/glossary/midroll/]

MOV—The Apple QuickTime media file format that was the basis for the MPEG-4 ISO base media file format. (ISO/IEC 14496-12, MPEG-4 Part 12)

MP2—An acronym for the MPEG 1/2 Layer II codec, MP2 is a lossy audio compression format first defined by ISO/IEC 11172-3:1993 (MPEG-1 Part 3) and then extended in ISO/IEC 13818-3:1998 (MPEG-2 Part 3)

to accommodate additional bitrates, sample rates, and up to 5.1 multichannel support. MP2 is based on a 32-subband transform, driven by a psychoacoustic model. It was primarily designed for Digital Audio Broadcasting and digital TV, but also found a place in “contribution” (i.e. links and/or content storage within broadcasting and streaming facilities prior to transmission to the final consumer) because it can be perceptually transparent at high bitrates. Coding efficiency is lower than MP3, and perceptual quality for broadcast and streaming becomes overtly unacceptable at bitrates below 128 kbps. The patents covering this are now expired.

MP3—An acronym for MPEG 1 and 2, Layer III, MP3 is a popular audio coding format for lossy digital audio compression developed by Fraunhofer and Bell Labs and standardized in ISO/IEC 11172-3:1993 (MPEG-1 Part 3) and then extended in ISO/IEC 13818-3:1998 (MPEG-2 Part 3) and ISO/IEC 23000-2:2008 for the MPEG Music Player format. It transforms audio between the time domain and the frequency domain. The encoder uses a psychoacoustic model to estimate whether quantization noise will be audible in each frequency-domain subband and allocates the number of bits used to code that subband accordingly. Compared to MP2’s 32 subbands, MP3 uses a large number of subbands in the frequency domain to improve coding efficiency: 576 subbands for non-transient material and 192 subbands for transient material. Switching the number of subbands limits the temporal spread of quantization noise accompanying the transient. MP3 is not 100% perceptually transparent at any bitrate and the quality degrades objectionably below 128 kbps. While still widely used, it has been technologically superseded by AAC. The patents covering MP3 are now expired.

MP4—ISO MPEG container format, standardized in ISO/IEC 14496-12:2022 and ISO/IEC 14496-14:2020 which defines the structure for time-base file format. Originally Apple QuickTime MOV, MP4 is used for high-quality video or audio. Different audio and video codecs may be used with the MP4 container. Derivatives of this container are M4V, M4A, and M4S.

MPEG—An acronym for the Moving Picture Experts Group, a standards-generating body under ISO/IEC JTC 1/SC29.

MPEG-2—A group of international standards. Within MPEG-2, the MPEG-2 Audio section as defined in Part 3 (ISO/IEC 13818-3: 1998) of the standard, enhances MPEG-1's audio by allowing the coding of audio programs with more than two channels, up to 5.1 multichannel. This method is backwards-compatible (also known as MPEG-2 BC), allowing MPEG-1 audio decoders to decode the two main stereo components of the presentation. MPEG-2 part 3 also defined additional bit rates and sample rates for MPEG-1 Audio Layer I, II and III.

MPEG-4—A group of international standards for the compression of digital audio and visual data, multimedia systems, and file storage formats. It was originally introduced in late 1998 as a group of audio and video coding formats and related technology agreed upon by the ISO/IEC Moving Picture Experts Group (MPEG) (ISO/IEC JTC1/SC29/WG11) under the formal standard ISO/IEC 14496—Coding of audio-visual objects. Uses of MPEG-4 include compression of audiovisual data for internet video and CD distribution, voice (telephone, videophone) and broadcast television applications.

MPEG-4 Audio Lossless Coding (MPEG-4 ALS)—An extension to the MPEG-4 Part 3 audio standard to allow lossless audio compression. The extension was finalized in December 2005 and published as ISO/IEC 14496-3:2005/Amd 2:2006 in 2006. The latest description of MPEG-4 ALS was published as subpart 11 of the MPEG-4 Audio standard (ISO/IEC 14496-3:2019) (5th edition) in December 2019.

MPEG-A—A group of standards for composing MPEG systems formally known as ISO/IEC 23000 - Multimedia Application Format, published since 2007. It consists of 20 Parts.

MPEG-D—A group of standards for audio coding formally known as ISO/IEC 23003—MPEG audio technologies. It consists of five parts including MPEG-D Part 3—Unified speech and audio coding (ISO/IEC 23003-3) and MPEG-D Part 4, Dynamic Range Control (ISO/IEC 23003-4). See USAC.

MP4 Box—A substructure of the extensible MP4 ISO BMFF hierarchical file structure. The file structure can be thought of as an outline form and can be examined with Box viewer software.

MPEG Common Media Application Format (CMAF)—A media container format optimized for large scale delivery of a single encrypted, adaptable multimedia presentation to a wide range of devices and adaptive streaming methods, including HTTP Live Streaming [RFC8216] and MPEG-DASH [ISO/IEC 23009-1:2022].

It is based on the ISO BMFF and supports the AAC, FLAC, ALAC, AVC and HEVC codecs and subtitles using IMSC1 and **WebVTT**. Its goal is to reduce media storage and delivery costs by using a single common media format across different client devices. [https://www.w3.org/TR/media-timed-events/#mpeg-common-media-application-format-cmaf]

MPEG-DASH—See Dynamic Adaptive Streaming over HTTP. [ISO/IEC 23009-1:2022]

MPX-FM—In analog FM broadcast, a widely standardized method of transmitting stereo on one FM radio channel. It transmits the sum of the left and right channels in the baseband below 19 kHz and the difference of the left and right channels on a 38 kHz amplitude modulated suppressed-carrier subcarrier. A 19 kHz pilot tone allows the receiver to regenerate the 38 kHz subcarrier accurately. With proper receiver design, MPX-FM can coexist with digital carriers on the same RF channel, a technology known as IBOC (in-band on-channel).

Multi-Bitrate (MBR) Streaming—MBR allows broadcasters to offer a wide range of stream qualities to improve the overall Quality of Experience. With multi-bitrate streaming, the audience can choose the highest quality stream that their connection and device can handle without interruptions. This technology, paired with **Adaptive Bitrate** streaming, creates the ideal setup to produce an optimal Quality of Experience for all listeners. HLS and MPEG-DASH support MBR; Icecast and SHOUTcast do not.

Network Time Protocol (NTP)—A networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks. It is intended to synchronize all participating computers to within a few milliseconds of Coordinated Universal Time (UTC). NTP can usually maintain time to within tens of milliseconds over the public internet, and can achieve better than one millisecond accuracy in local area networks under ideal conditions, although asymmetric routes and network congestion can cause errors of 100 ms or more. The protocol is usually described in terms of a client–server model, but can as easily be used in peer-to-peer relationships where both peers consider the other to be a potential time source. NTP is standardized in RFC 5905 by the Internet Engineering Taskforce (IETF).

Now-Playing Information—Metadata intended for display by the player client. It almost always includes the title of the audio content and artist information. It may include other display elements like album art.

Ogg—A free, open container format maintained by the Xiph.Org Foundation. The authors of the Ogg format state that it is unrestricted by software patents and is designed to provide for efficient streaming and manipulation of high-quality digital multimedia.

Ogg Opus—see **Opus Audio Codec**.

Ogg Vorbis—see **Vorbis Audio Codec**.

On-demand—On-demand/file streaming is playback pulled from existing recorded content at the player's request. As opposed to live streaming, where the player client receives content pushed to it by the server after connection, with on-demand streaming the player controls the timing of the content to be played out. On-demand streaming is also known as File-based Streaming or Podcasting.

Open Standard—A standard made available (sometimes for a fee) to the general public and developed (or approved) and maintained via a collaborative and consensus-driven process. Open standards facilitate interoperability and data exchange among different products or services and are intended for widespread adoption. Open standards are not to be confused with open-source software.

Open-Source—Any computer program or software component whose source code is made available for use or modification as users or other developers see fit. Unlike proprietary software, open source software is developed as a public, open collaboration and made freely available to the public. There are several usage and distribution licensing protocols relating to open-source software, which may require those distributing software that includes open-source elements to do specified things like disclosing source code that the authors have added.

Opus Audio Codec—An open source, royalty-free, lossy codec standardized by the Internet Engineering Task Force (IETF) as RFC 6716. It is usually packaged in the Ogg container format and is called "Ogg Opus."

OTA (Over the Air)—Refers to content delivered to consumers via antenna-received RF broadcasts from licensed broadcast stations. Cellular connections to streaming providers, while also conveyed via RF, are not considered “over-the-air” in this context.

OTT (Over-the-Top)—In video streaming, refers to content streamed to consumer set-top receiver boxes via the internet.

Out-of-Band—Two or more sets of data sent in separate transports and/or containers. Usually refers to audio and associated PAD/metadata. Out of band metadata is not synchronous.

Page Impression—An analytical term conveying the number of hits that a stream or page receives.

PHP—An open source, server-side, interpreted, dynamically-typed scripting language designed for web development and can be embedded into HTML. It features a simple and easy-to-learn syntax, with support for object-oriented programming and functional programming paradigms. PHP is typically used in conjunction with a web server, such as Apache, to generate dynamic web pages and to interact with databases, such as MySQL or PostgreSQL. PHP also includes a large and growing number of built-in functions and libraries for common web development tasks, such as handling forms, processing data, and sending email. Additionally, PHP can be used for command-line scripting and supports multiple platforms, including Windows, Linux, and macOS. PHP was originally an abbreviation of Personal Home Page but it now stands for the recursive initialism PHP: Hypertext Preprocessor. The PHP reference implementation is now produced by The PHP Group. [https://www.php.net/manual/en/intro-whatis.php]

Player—A client application or appliance that allows an audience to consume a decoded stream.

Playlist—M3U8—HLS—UTF8—A file that provides instructions to a player so it can assemble segments or chunks for continuous playout. The playlist may be dynamic for live content or static for file/on-demand content. HLS and MPEG-DASH use this technology to achieve the advantages discussed in Section 5.1.

Playout System—A system running software that than can, under control of a pre-programmed playlist or live operator, sequentially play out content and metadata from local or networked media to assemble a continuous stream or broadcast. In live assist, the output of the playout system may be mixed with other elements like live speech before streaming or broadcast. See Playout Systems.

PNG—See **Portable Network Graphics**.

Podcast—An on-demand digital audio file or set of files that can be fetched over a network by a personal device and/or player. The most common file format is simple mp3 or mp4/m4a; however for advanced rich-media presentations, HLS segments can be used and can contain dynamic metadata and images, which require a special encoder. See **On-demand**.

Polling metadata—An inefficient, asynchronous way to convey metadata from a server to a client, where the client makes repeated requests to the server for data at a specified time interval.

Portable Network Graphics (PNG)—A raster-graphics file format that supports lossless data compression. The PNG working group designed the format for transferring images on the internet, not for professional-quality print graphics; therefore, non-RGB color spaces such as CMYK are not supported. A PNG file contains a single image in an extensible structure of chunks, encoding the basic pixels and other information such as textual comments and integrity checks documented in RFC 2083. PNG files have the “.png” file extension and the “image/png” MIME media type. PNG was published as an informational RFC 2083 in March 1997 and as an ISO/IEC 15948 standard in 2004. [Wikipedia]

Portable People Meter (PPM)—A Nielsen audience analytics system that measures a sample of the broadcast or streaming audience. The PPM is worn like a pager and detects hidden audio tones (“watermarks”) injected within the streamed or transmitted content, logging each time it finds a signal. The watermark is encoded metadata that identifies the stream or broadcast. It is embedded in the audio content of the broadcast or stream using a psychoacoustic masking model that provides a program-dependent watermark level, varying the amount of watermark injection so that it is psychoacoustically masked by the content and thus inaudible to listeners. The PPM contains a motion sensor to estimate if a listener is actively engaged with the content.

Precision Time Protocol (PTP)—A protocol used to synchronize clocks throughout a computer network. On a local area network, it achieves clock accuracy in the sub-microsecond range, making it suitable for measurement and control systems. The first version of PTP, IEEE 1588-2002, was published in 2002. IEEE 1588-2008, also known as PTP Version 2 is not backward compatible with the 2002 version. IEEE 1588-2019 was published in November 2019 and includes backward-compatible improvements to the 2008 publication. IEEE 1588-2008 includes a profile concept defining PTP operating parameters and options. Several profiles have been defined for applications including telecommunications, electric power distribution and audiovisual. IEEE 802.1AS is an adaptation of PTP for use with Audio Video Bridging and Time-Sensitive Networking.

Pre-Roll–An online term describing interstitials inserted before the beginning of a program. See Mid-Roll.

Program Associated Data (PAD)—Metadata specific to program or now-playing content. Also referred to as Program Service Data (PSD). PAD is the data that is displayed by streams, HD Radio, DRM (Digital Radio Mondiale), and satellite radio players. This is sometimes referred to as "Now-Playing Information." This provides information to the listener, such as song title, artist, genre, and/or station slogan. This is different from Radio Data System (RDS).

Program Associated Graphics (PAG)—A specific form of Program Associated Data that allows a compatible player to display graphical elements like album art.

Progressive Download—Content that plays immediately upon download. This is a fundamental feature and one of several advantages of segmented streaming such as HLS and MPEG-DASH.

Pulse Code Modulation (PCM)—A method used to digitally represent sampled analog signals. It is the standard form of digital audio in computers, compact discs, digital telephony and other digital audio applications. In a PCM stream, the amplitude of the analog signal is sampled at uniform intervals and each sample is quantized to the nearest value within a range of digital steps. Linear pulse-code modulation (LPCM) is a specific type of PCM in which the quantization levels are linearly uniform. This is in contrast to PCM encodings in which quantization levels vary as a function of amplitude (as with the A-law algorithm or the μ-law algorithm). Though PCM is a more general term, it is often used to describe data encoded as LPCM. A PCM stream has two basic properties that determine the stream's fidelity to the original analog signal: the sampling rate, which is the number of times per second that samples are taken; and the bit depth, which determines the number of possible digital values that can be used to represent each sample.

In streaming, PCM is often used informally to describe a signal that has not been subject to lossy compression, even though the audio usually was initially in PCM form. In contrast to PCM, a small minority of content was released to consumers in the Super Audio CD (SACD) format. SACD audio is stored in Direct Stream Digital (DSD) format using pulse-density modulation (PDM) where audio amplitude is determined by the varying proportion of 1s and 0s. For almost all streaming codecs, DSD audio must be converted to PCM before encoding.

QR (Quick Response) Code—A type of two-dimensional matrix barcode. A barcode is a machine-readable optical image that contains information specific to the labelled item. In practice, QR codes contain data for a locator, an identifier, and website visitor tracking. To efficiently store data, QR codes use four standardized modes of encoding (i) numeric, (ii) alphanumeric, (iii) byte or binary, and (iv) kanji.[3]. Consumer smartphones typically include QR-reading software that allows the phone to act on the content of the code such as opening a webpage in the phone's browser.

Quality-of-Service (QoS)—The description or measurement of the overall performance of a service, such as a telephony or computer network, or a cloud computing service, particularly the performance seen by the users of the network. To quantitatively measure quality of service, several related aspects of the network service are often considered, such as packet loss, bit rate, throughput, transmission delay, availability, jitter, etc. In the field of computer networking and other packet-switched telecommunication networks, quality of service refers to traffic prioritization and resource reservation control mechanisms rather than the achieved service quality. Quality of service is the ability to provide different priorities to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow.

RAW—With reference to media files, audio data without a container and metadata tagging. For example, an MP4 file can contain raw AAC data along with other elements like metadata.

RDS-UECP—Radio Data System–Universal Encoder Communication Protocol. RDS is the metadata system used in FM analog broadcast to present information to the listener and to control the receiver, allowing it (for example) to automatically tune to a different station carrying the same content when the receiver gets out of reach of the station to which it is currently tuned. UECP is specified in Part 10 of the RDS standard. Note that it does not. include Scrolling PS, which a popular extension that must be implemented as a manufacturer specific command or rolled in software. See RDS for more detail.

RDS Radio Text (RT)—In the RDS metadata bitstream, a 128 (maximum) character message to be displayed by the receiver if so equipped.

Redirect—URL redirection, also known as URL forwarding, is a technique to give more than one URL address to a page, a form, or a whole Web site/application. HTTP has a special kind of response, called a HTTP redirect, for this operation. In HTTP, redirection is triggered by a server sending a special redirect response to a request. Redirect responses have status codes that start with 3, and a Location header holding the target URL. When browsers receive a redirect, they immediately load the new URL provided in the Location header. Besides the small performance hit of an additional round-trip, users rarely notice the redirection. [https://developer.mozilla.org/en-US/docs/Web/HTTP/Redirections]

ReplayGain—A proposed technical standard published by David Robinson in 2001 to measure and normalize the perceived loudness of audio in computer audio formats such as MP3 and Ogg Vorbis. It allows media players to normalize loudness for individual tracks or albums. The original specification was later revised to use the ITU-R BS.1770-3 algorithm for loudness measurement.

Request for Comments (RFC)—A publication in a series from the principal technical development and standards-setting bodies for the internet, most prominently the **Internet Engineering Task Force** (IETF). An RFC is authored by individuals or groups of engineers and computer scientists in the form of a memorandum describing methods, behaviors, research, or innovations applicable to the working of the internet and internet-connected systems. It is submitted either for peer review or to convey new concepts, information, or, occasionally, engineering humor.

Real Time Messaging Protocol (RTMP)—An Adobe proprietary protocol that maintains persistent connections and also allows communication with low latency for streaming media delivery. This was part of Adobe Flash and is now deprecated for several security and compatibility reasons.

Resource Interchange File Format (RIFF)[12]—A tagged file structure for multimedia resource files. Strictly speaking, RIFF is not a file format, but a file structure that defines a class of more specific file formats,. The basic building block of a RIFF file is called a chunk. Chunks are identified by four-character codes and an application such as a player will skip chunks with codes it does not recognize. The basic chunk is a RIFF chunk, which must start with a second four-character code, a label that identifies the particular RIFF "form" or subtype. Applications that play or render RIFF files may ignore chunks with labels they do not recognize. Chunks can be nested. The RIFF structure is the basis for a few important file formats including wav, avi and webp, but has not been used as the wrapper structure for any new file formats developed since the mid-1990s.

RTSP/RTP—RTSP is a presentation-layer protocol that lets end users command media servers via pause and play capabilities. RTP is the transport protocol used to move said data. [wowza.com]

Scalable Vector Graphics (SVG)—An XML-based vector image format for defining two-dimensional graphics, having support for interactivity and animation. The SVG specification is an open standard developed by the World Wide Web Consortium since 1999. SVG images are defined in a vector graphics format and stored in XML text files. SVG images can thus be scaled in size without loss of quality, and SVG files can be searched, indexed, scripted, and compressed. The XML text files can be created and edited with text editors or vector graphics editors, and are rendered by the most-used web browsers.

---

[12] As of this writing, the following sites contain useful information on RIFF:
https://learn.microsoft.com/en-us/windows/win32/multimedia/resource-interchange-file-format-services
https://www.loc.gov/preservation/digital/formats/fdd/fdd000025.shtml
http://soundfile.sapp.org/doc/WaveFormat;
https://docs.fileformat.com/audio/wav/

SCTE-35 (ANSI/SCTE 35 2019r1)—This standard, “Digital Program Insertion Cueing Message for Cable” (SCTE 35), is the core signaling standard for advertising and distribution control (e.g., blackouts) of content for content providers and content distributors. See section 11.1.

Securable Reliable Transport (SRT)—An open-source, codec agnostic, media transport protocol used for content ingest or point-to-point. It is an improvement and replacement for deprecated Adobe Flash RTMP, as RTMP is not codec agnostic and does not support modern codecs.

Secure Real Time Protocol (SRTP)—An extension profile of RTP (Real-Time Transport Protocol) that adds further security features, such as message authentication, confidentiality and replay protection mostly intended for VoIP communications. [https://www.3cx.com/voip/srtp/]

Secure Shell Protocol (SSH)—The Secure Shell Protocol (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network. Its most notable applications are remote login and command-line execution.

SSH applications are based on a client–server architecture, connecting an SSH client instance with an SSH server. SSH operates as a layered protocol suite comprising three principal hierarchical components: the transport layer provides server authentication, confidentiality, and integrity; the user authentication protocol validates the user to the server; and the connection protocol multiplexes the encrypted tunnel into multiple logical communication channels.

The protocol specification distinguishes two major versions, referred to as SSH-1 and SSH-2. The most commonly implemented software stack is OpenSSH, released in 1999 as open-source software by the OpenBSD developers. SSH does not have an RFC. Implementations are distributed for all types of operating systems in common use, including embedded systems. SSH is used for SFTP.

Secure Socket Layer (SSL)/Transport Layer Security (TLS)—Standard technologies designed to ensure the security of internet connections and protect sensitive data during transmission between systems. These protocols prevent unauthorized access and manipulation of information, including personal details. SSL/TLS can be employed between various entities, such as a server and client, a website and browser, or even between servers. While widely used in applications like email, instant messaging, and voice over IP, their most visible application is securing HTTPS connections.

SSL achieves data confidentiality by encrypting the transmitted data. It employs cryptographic algorithms to ensure data security during transit.

Transport Layer Security (TLS) serves as an enhanced successor to SSL, sharing similar encryption mechanisms for safeguarding data and information. While the terms SSL and TLS are often used interchangeably, SSL remains the most commonly used term.

TLS primarily focuses on providing security, including confidentiality, integrity, and authenticity, by utilizing cryptographic techniques such as certificates between communicating computer applications. The protocol operates within the presentation layer and consists of two layers: the TLS record and the TLS handshake protocols.

Datagram Transport Layer Security (DTLS) is closely related to TLS and provides security for datagram-based applications. In technical writing, references to "(D)TLS" are commonly used to denote both versions.

TLS is an Internet Engineering Task Force (IETF) standard, initially defined in 1999. The current version is TLS 1.3, defined in RFC 8446 (August 2018). TLS builds upon the deprecated SSL specifications (1994, 1995, 1996) developed by Netscape Communications to introduce the HTTPS protocol into their Navigator web browser.

Segment—A portion or chunk of a media stream of a defined length used for HLS and MPEG/DASH streaming protocols. Several segments make up the live or file (on-demand) stream.

Server-Side Ad Insertion (SSAI)—Advertising inserted into the stream at the streaming server, downstream from studio playout. Similar to “local ad insertion” by network affiliates in broadcast.

SHOUTcast—A closed streaming platform and server using the original ICY protocol that allows anyone to stream audio over the internet. It was formerly owned by Nullsoft/AOL. ICY is a proprietary modification of the HTTP protocol requiring specialized servers and was never formally standardized. It has gained huge

acceptance because of its simplicity. ICY metadata is asynchronous and supports a limited feature set. It is carried in the natively supported StreamTitle field, which is a concatenation of the Title and Artist fields. It does not use ID3 metadata.

Silverlight™—A proprietary Microsoft application framework for writing and running rich internet applications, with features and purposes similar to those of Adobe Flash. The run-time environment for Silverlight was available as a plug-in for most web browsers. Silverlight is now deprecated due to security vulnerabilities. No current browser supports it.

Simple Network Management Protocol (SNMP)—A standards-based protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change device behavior. As of this writing, the current secure SNMP is v3. Linux supports SNMP v3. Microsoft operating systems do not offer v3 without third-party software and are therefore subject to more security vulnerabilities.

SNMP—See **Simple Network Management Protocol**.

SoundExchange—An American non-profit collective rights management organization. It is the sole organization designated by the U.S. Congress to collect and distribute digital performance royalties for sound recordings. This is how artists get paid from broadcast and streaming music services. Generally, a statutory licensee must provide information monthly to SoundExchange to identify which tracks have been used under such licensing. This information includes artist, title and International Standard Recording Code ("ISRC"). SoundExchange pays featured and non-featured artists and master rights owners for the non-interactive use of sound recordings under the statutory licenses set forth in 17 U.S.C. § 112 and 17 U.S.C. § 114.

SoundExchange handles the following duties with respect to statutory licenses: Collects performance royalties from the statutory licensees collects and processes all data associated with the performance of the sound recordings; allocates royalties for the performance of the sound recording based on all of the data collected and processed; distributes the featured artist's share directly to the artist; distributes the copyright owner's share directly to the copyright owner; distributes the non-featured artists' shares to SAG-AFTRA and AFM's Intellectual Property Rights Distribution Fund; and provides detailed reports summarizing the titles, featured artists, and royalty amounts for each of the sound recordings performed by the statutory licensees.

The report generator uses metadata to create the report. Much of this work can be done by the CDN but the streamer is ultimately responsible. Periodically, the reports that are extracted from metadata will be compiled and then audited by SoundExchange to determine if the streamer is paying the correct amount.

SSH File Transfer Protocol (SFTP)—A network protocol that provides file access, file transfer, and file management over any reliable data stream. It was designed by the Internet Engineering Task Force (IETF) as an extension of the Secure Shell protocol (SSH) version 2.0 to provide secure file transfer capabilities, typically over port 22. The IETF Internet Draft states that even though this protocol is described in the context of the SSH-2 protocol, it could be used in a number of different applications such as secure file transfer over Transport Layer Security (TLS) and transfer of management information in VPN applications. The protocol is designed to be run over a secure channel such as SSH, where the server has already authenticated the client, and where the identity of the client user is available to the protocol. SFTP is not FTP run over SSH, but rather a new protocol designed from the ground up by the IETF SECSH working group.

SSL Certificates—A digital cryptographic credential, or digital certificate, that facilitates authenticating the identity of a website and creating an encrypted connection for SSL/TLS protocols. It contains information such as the website owner's corporate name, the issuing authority, and other relevant details, safeguarding sensitive data transmission and enhancing the overall security of online interactions.

Security certificates are employed to secure online communications. While the term "SSL certificate" is more widely used, it now generally encompasses the updated Transport Layer Security protocol. See Section **Error! Reference source not found.**.

To view specific information about the certificate, users can click on the lock symbol typically found in the URL bar of most web browsers.

Stateful Packet Inspection—A security feature in a stateful network firewall. It is also referred to as dynamic packet filtering. A stateful firewall keeps track of the state of network connections, such as TCP streams, UDP datagrams, and ICMP messages, and can apply labels such as LISTEN, ESTABLISHED, or CLOSING. State table entries are created for TCP streams or UDP datagrams that are allowed to communicate through the firewall in accordance with the configured security policy.

TEMPLATE—[Methods for metadata export and import]

TEMPLATE ASSIGN—Metadata Formatting as found in the following workflow: FILE > PLAYOUT > TEMPLATE > NOW PLAYING > NETWORK > TEMPLATE > STREAM ENCODER

Timed Text Markup Language (TTML)—Textual information that may be used directly as a distribution format for online captioning and subtitles, and as an interchange format among legacy distribution content formats. It is a superset that encompasses preceding captioning approaches. It supports the semantics of most of the legacy closed caption files. It is used in the video industry for the purpose of authoring, transcoding and exchanging timed text information and for delivering captions, subtitles. It is a text format, based on multiple W3C technologies such as XML, SMIL for the timing semantics, and XSL-FO and CSS for presentation and layout of text. Typical uses of timed text are the realtime subtitling of foreign-language movies on the Web, captioning for people lacking audio devices or having hearing impairments, karaoke, scrolling news items, and teleprompter applications.
[https://www.w3.org/AudioVideo/TT/docs/TTML-Profiles.html]

Time Spent Listening (TSL)—An analytical term representing an estimate, expressed in hours and minutes, of the amount of time the average listener spent with a given content source during a particular daypart.

Total Line Reporting (TLR)—For stations that opt-in to the service and meet Nielsen's requirements, a Nielsen ratings reporting method applicable when a given broadcast station’s stream and terrestrial over-the-air product carry identical content. It is used when the stream and over-the-air services are not separately monetized.

Transcoding –Taking audio encoded with a given codec and re-encoding it using the same codec at a different bitrate or using a different codec (e.g. transcoding MP3 to HE-AACv2).. While the most straightforward method is to decode the audio into linear PCM and then to re-encode it, it is sometimes possible to transcode without a full decode.

Transpacketization—Taking content conveyed in a given container and inserting it into a different container (e.g., transpacketizing SHOUTcast to HLS). If the same codec and bitrate are used in the target container, this process does not require transcoding the audio.

Transport Format—Describes the container format in which the stream is packaged (e.g. HLS, Icecast, RTMP) and whether the metadata is or is not in-band.

Transport Layer Security (TLS)—See Secure Socket Layer (SSL)/Transport Layer Security (TLS).

Triton Analytics—A system that estimates the number of listeners to a stream by counting the number of connections between the server and player.

TS (Transport Stream)—A standard digital container format for transmission and storage of audio, video, and Program and System Information Protocol (PSIP) data. It is used in broadcast systems such as DVB, ATSC and IPTV. Transport stream specifies a container format encapsulating packetized elementary streams, with error correction and synchronization pattern features for maintaining transmission integrity when the communication channel carrying the stream is degraded. [Wikipedia]

TTML—See Timed Text Markup Language.

TCP/IP—The internet protocol suite, commonly known as TCP/IP, is the set of communication protocols used in the internet and similar computer networks. The current foundational protocols in the suite are the Transmission Control Protocol (TCP) and the Internet Protocol (IP), as well as the User Datagram Protocol (UDP). The internet protocol suite provides end-to-end data communication specifying how data should be packetized, addressed, transmitted, routed, and received. This functionality is organized into four abstraction layers, which classify all related protocols according to each protocol's scope of networking.

From lowest to highest, the layers are the link layer, containing communication methods for data that remains within a single network segment (link); the internet layer, providing internetworking between independent networks; the transport layer, handling host-to-host communication; and the application layer, providing process-to-process data exchange for applications. [Wikipedia]

UDP/IP—In computer networking, the User Datagram Protocol (UDP) is one of the core communication protocols of the internet protocol suite used to send messages (transported as datagrams in packets) to other hosts on an Internet Protocol (IP) network. Within an IP network, UDP does not require prior communication to set up communication channels or data paths. UDP uses a simple connectionless communication model with a minimum of protocol mechanisms. [Wikipedia]

Unified Speech and Audio Coding (USAC)—A lossy audio codec particularly suited for low bit rate streaming using AMR Wideband, HE-AACv2, HE-AACv1 and AAC-LC, depending upon bitrate, in one single audio container. Compared to HE-AACv2, it provides improved music quality at very low bitrates due to incorporation of enhanced variations of the MPEG-4 Spectral Band Replication (SBR) and MPEG-D MPEG Surround parametric coding tools, and improved speech quality due to incorporation of time-domain linear prediction and residual coding tools (ACELP-like techniques). Standardized in ISO/IEC 23003-3, MPEG-D Pt.3. The MPEG4 Audio Object type is in ISO/IEC 14496-3:2020. See Section 12.2, USAC/xHE-AAC™.

Unicode—An information technology standard for the consistent encoding, representation, and handling of text expressed in most of the world's writing systems. The standard, which is maintained by the Unicode Consortium, defines as of the current version (15.0) 149,186 characters covering 161 modern and historic scripts, as well as symbols, emoji (including in colors), and non-visual control and formatting codes. As of this writing, it is standardized in the document "The Unicode Consortium. The Unicode Standard, Version 15.0.0, (Mountain View, CA: The Unicode Consortium, 2022. ISBN 978-1-936213-32-0)." https://www.unicode.org/versions/Unicode15.0.0/

Universal Coordinated Time (UTC)— The primary time standard by which the world regulates clocks and time. Time zones around the world are expressed using positive or negative offsets from UTC. UTC is derived from International Atomic Time by adding or subtracting an integer number of leap seconds to correct for the Earth's rotation. The number of leap seconds is chosen so that mean solar noon at the Greenwich meridian does not deviate from UTC noon by more than 0.9 seconds.

Universal Resource Identifier (URI)—A unique sequence of characters that identifies a logical or physical resource used by web technologies. URIs may be used to identify anything, including real-world objects, such as people and places, concepts, or information resources such as web pages and books. Some URIs provide a means of locating and retrieving information resources on a network (either on the internet or on another private network, such as a computer filesystem or an Intranet); these are Uniform Resource Locators (URLs). A URL provides the location of the resource. A URI identifies the resource by name at the specified location or URL. Other URIs provide only a unique name, without a means of locating or retrieving the resource or information about it, these are Uniform Resource Names (URNs). The web technologies that use URIs are not limited to web browsers. URIs are used to identify anything described using the Resource Description Framework (RDF), for example, concepts that are part of an ontology defined using the Web Ontology Language (OWL), and people who are described using the Friend of a Friend vocabulary would each have an individual URI.

Uniform Resource Locator (URL)—Colloquially termed a web address, a URL is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. A URL is a specific type of Universal Resource Identifier (URI) although many use the two terms interchangeably. URLs occur most commonly to reference web pages (HTTP/HTTPS) but are also used for file transfer (FTP), email (mailto), database access (JDBC), and many other applications.

URL Encoding—A method to encode arbitrary data in a Universal Resource Identifier (URI) using only the limited US-ASCII characters legal within a URI. Although it is known as URL encoding, it is also used more generally within the main Uniform Resource Identifier (URI) set, which includes both Uniform Resource Locator (URL) and Uniform Resource Name (URN). As such, it is also used in the preparation of data of the application/x-www-form-urlencoded media type, as is often used in the submission of HTML form data in HTTP requests. This is not to be confused with HTML5 Encoding. For examples, see URL Encoding.

URL Query String—Part of a Uniform Resource Locator(URL) that assigns values to specified parameters. A query string commonly includes fields added to a base URL by a Web browser or other client application, for example as part of an HTML document, choosing the appearance of a page, jumping to positions in multimedia content, or sending metadata to a server.

e.g.: https://www.example.com/directory/file.php?title=A%20For%20You

Note that URL Query Strings are URL Encoded UTF-8.

The query string is composed of a series of field-value pairs.

e.g.: field1=value1&field2=value2&field3=value3...

A URL Query String may also invoke a small program or process on the web server for a particular function.

UTF-8—A variable-width character encoding used for electronic communication. Defined by the Unicode Standard, the name is derived from Unicode (or Universal Coded Character Set) Transformation Format—8-bit. [Wikipedia]

UTF8mb3—UTF-8mb3 is a UTF-8-derived character encoding that uses 1 to 3 bytes to depict one code point. Hence, the character set “utf8”/”utf8mb3” cannot depict all Unicode code points; it only supports the range 0x000 to 0xFFFF, which is called the “Basic Multilingual Plane.” [dev.mysql.com/]

UTF8mb4—UTF-8mb 4 is a UTF-8-derived character encoding that uses 1 to 4 bytes to depict one code point, so it can depict the entire Unicode character set. [dev.mysql.com/]

UTF-16—(16-bit Unicode Transformation Format) is a character encoding capable of encoding all valid code points of Unicode. The encoding is variable-length, as code points are encoded with one or two 16-bit code units. [Wikipedia]

Vorbis Audio Codec -An audio coding format and software reference encoder/decoder (codec) for lossy audio compression. Vorbis is most commonly used in conjunction with the Ogg container and it is therefore often called Ogg Vorbis.

Video Ad Serving Template (VAST)—A template for structuring ad tags that serve video and audio ads to media players. Using an XML schema, VAST transfers important metadata about an ad from the ad server to a media player. Launched in 2008, VAST has since played an important role in the growth of the digital video and audio marketplace. [https://iabtechlab.com/wp-content/uploads/2022/09/VAST_4.3.pdf]

Wasm—see **WebAssembly**.

WAV—Microsoft audio container format based on the RIFF file format, usually used for linear PCM audio in various formats, but not limited to that—it may contain compressed audio, sometimes even MP3.WAV files can contain native metadata in their List INFO chunk. Custom metadata chunks can be used, including ID3.

WAV List INFO—RIFF INFO tags found in AVI video and WAV audio files The EXIF 2.3 specification defines 89 tags such as DateCreated and Genre. Other tags are found in AVI files generated by some software. List INFO is the native metadata format for RIFF WAV files. WAV List INFO is limited to ASCII characters; UTF8 is not supported. [https://exiftool.org/TagNames/RIFF.html#Info]

WebAssembly (Wasm)—A binary instruction format for a stack-based virtual machine. Wasm is a portable compilation target for programming languages, enabling deployment on the web for client and server applications. The Wasm stack machine is designed to be encoded in a size- and load-time-efficient binary format. WebAssembly aims to execute at native speed by taking advantage of common hardware capabilities available on a wide range of platforms. It can be used where JavaScript is too slow for certain applications, and/or for implementing DSP and codecs that are unavailable in operating systems. https://webassembly.org/

WebAudio—The Web Audio API provides a powerful and versatile system for controlling audio on the Web, allowing developers to choose audio sources and destinations, add effects to audio, create audio visualizations, apply spatial effects, mixing, routing, streaming players, and much more. https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

Web Distributed Authoring and Versioning (WebDAV)—See **Distributed Authoring and Versioning**.

Web RTC—A free and open-source project providing web browsers and mobile applications with real-time communication via application programming interfaces (APIs). It is supported on native clients for all major platforms, and on all major browsers through JavaScript APIs.

It allows audio, video and text communication to work inside web pages by allowing direct peer-to-peer communication, eliminating the need to install plugins or download native apps. It is used for low-latency point-to-point audio and video connections and is the foundation for conferencing and remote media connectivity.

Web RTC is not for streaming to large audiences due to its limited reliability and client capacity. It uses UDP. WebRTC specifications have been published by the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF).

Web Video Text Tracks (WebVTT)— A text track format whose main use is for marking up external text track resources in connection with the HTML <track> element. WebVTT files provide captions or subtitles for video content, and also text video descriptions, chapters for content navigation, and more generally any form of metadata that is time-aligned with audio or video content. Its specification, found at https://www.w3.org/TR/webvtt1/, is based on the Draft Community Group Report of the Web Media Text Tracks Community Group.

Wordclock—Although sometimes used generically to describe reference sample frequency distribution in a facility by hardware such AES11, wordclock most commonly refers to a squarewave at 1x the sample frequency (or sometimes power-of-two multiples thereof), distributed to hardware devices over coaxial cable terminated in BNC connectors.

xHE-AAC™— A trademark for a Fraunhofer implementation of MPEG-D USAC (ISO/IEC 23003-3:2020). It mandates inclusion of loudness and dynamic range control metadata, standardized in MPEG-D DRC (ISO/IEC 23003-4:2020) as amended.

XML—See **Extensible Markup Language**.

Annex A (Informative)
Player client: streaming server data communications

A.1 HLS

The HLS-fMP4 HTML5-MSE Player Client:Server Connection is described below.

There are three main steps to establishing an HLS fMP4 Player Client connection, displaying realtime metadata, and playing audio.

1. Connect and retrieve the Master or Variant .m3u8 file.
2. Parse .m3u8 from step (1) and connect and retrieve the initial init.mp4 segment.
3. Parse .m3u8 from step (1) and connect and retrieve successive m4s segments for the stream listening duration.

These three steps are illustrated in the tables below:

1. Example of a request URL from the player:
   https://domain.com/bucket/ram/00/01x/256k/program.m3u8

General

```
Request URL: https://domain.com/stream/256k/program.m3u8?ua=StreamS-MSE%2F1.0
&uuid=244a9c38-7b03-42f2-d745-ca27f653abf5
Request Method: GET
Status Code: 200
Remote Address: 12.34.56.78:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 473
content-type: application/x-mpegURL
date: Mon, 24 Apr 2023 22:14:29 GMT
etag: "4561df1ffa76d91:0"
last-modified: Mon, 24 Apr 2023 22:14:23 GMT
provider: $tream$ HiFi Radio
server: StreamS/2.0
```

Request Headers

```
DNT: 1
Referer: https://domain.com/player/js/hls-worker-min.js
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/111.0.0.0 Safari/537.36
```

2. Example of a reply from the server (contained in the m3u8 file):
   https://domain.com/bucket/ram/00/01x/256k/program_init.mp4

General

```
Request URL: https://domain.com/bucket/ram/00/01x/256k/program_init.mp4?StreamS-MSE-1.0_uuid=a78e5d3b-
c956-482b-d887-3f898db3b5ea
```

```
Request Method: GET
Status Code: 200
Remote Address: 123.45.67.89:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 662
content-type: video/mp4
date: Tue, 04 Oct 2022 23:20:05 GMT
etag: "a1a3183dc2cbd81:0"
last-modified: Mon, 19 Sep 2022 00:53:31 GMT
provider: $tream$ HiFi Radio
server: StreamServer HLS
```

Request Headers

```
DNT: 1
Referer: https://la2.indexcom.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/105.0.0.0 Safari/537.36
```

3. Example of a subsequent reply from the server (contained in the m3u8 file): https://domain.com/bucket/ram/00/01x/256k/program-20221004T231625Z.m4s?

General

```
Request URL: https://domain.com/bucket/ram/00/01x/256k/program-20221004T231625Z.m4s?StreamS-MSE-
1.0_uuid=a78e5d3b-c956-482b-d887-3f898db3b5ea
Request Method: GET
Status Code: 200
Remote Address: 24.199.52.152:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 322792
content-type: video/mp4
date: Tue, 04 Oct 2022 23:20:05 GMT
etag: "8381f8b247d8d81:0"
last-modified: Tue, 04 Oct 2022 23:19:06 GMT
provider: $tream$ HiFi Radio
server: StreamServer HLS
```

Request Headers

```
DNT: 1
Referer: https://la2.indexcom.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/105.0.0.0 Safari/537.36
```

HLS Server – Player Request – SSL HTTP/2
User-Agent: cURL (a command line application)

```
root@debian:~# curl -v https://www.domain.com/stream/program.m3u8
*   Trying 45.32.67.90:443…
* Connected to stream.domain.com (12.34.56.78) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  Cafile: /etc/ssl/certs/ca-certificates.crt
*  Capath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server accepted to use h2
* Server certificate:
*  subject: CN=streamindex.net
*  start date: Mar 22 18:46:58 2023 GMT
*  expire date: Jun 20 18:46:57 2023 GMT
*  subjectAltName: host “stream.domain.com” matched cert’s “*.stream.domain.com”
*  issuer: C=US; O=Let’s Encrypt; CN=R3
*  SSL certificate verify ok.
* Using HTTP2, server supports multi-use
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* Using Stream ID: 1 (easy handle 0x5571da691fe0)
> GET /stream/program.m3u8 HTTP/2
> Host: stream.domain.com
> user-agent: curl/7.74.0
> accept: */*
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Connection state changed (MAX_CONCURRENT_STREAMS == 8)!
< HTTP/2 200
< content-type: application/x-mpegURL
< content-length: 148
< accept-ranges: bytes
< cache-control: no-cache, no-store
< access-control-allow-origin: *
< access-control-allow-methods: GET, HEAD
< x-robots-tag: noindex, noarchive
< provider: StreamS HiFi
< server: StreamS/2.0
< date: Sun, 09 Apr 2023 07:32:12 GMT
<
#EXTM3U
#ENCODER:StreamS Live HTTP Encoder v3.2.1.183
#EXT-X-VERSION:6
#EXT-X-STREAM-INF:BANDWIDTH=134000,CODECS=”mp4a.40.2”
128k/program.m3u8
* Connection #0 to host bubblegum.la11.streamindex.net left intact
root@debian:~#
```

Until it is disconnected from the stream, the Player Client will continue to retrieve .m4s segments based on the encoder-specified segment size.

Note that there is no stream PAD or Now-Playing metadata exposed at this level. Metadata is contained in the .m4s segments and must be parsed out using HTML5-MSE JavaScript ISO BMFF Box and id3 frame tools. It is located in ISO BMFF emsg box as id3v2.4 frames and is UTF8mb4 encoded.

Audio is then passed to the operating system decoders using HTML5-MSE JavaScript.

The player client should generate a unique UUID/GUID for each session, which should be included in the HTTP GET Requests as a query string. This will help create a server log that can be parsed for accurate analytics and performance reporting. *Failure to do this can cause erroneous performance reporting, producing incorrect monetization calculations.*

The HTML5-MSE JavaScript to accomplish this is beyond the scope of this document; however, there is a free reference implementation available from StreamS/Modulation Index, LLC at https://www.indexcom.com/products/html5player/.

A.2 MPEG-DASH

The DASH-fMP4 HTML5-MSE Player Client:Server Connection is described below.

There are three main steps to establishing an DASH fMP4 Player Client connection, displaying realtime metadata, and playing audio.

1. Connect and retrieve the .mpd file.
2. Parse .mpd from step (1) and connect and retrieve init.mp4 segment.
3. Parse .mpd from step (1) and connect and retrieve successive .m4s segments for the stream listening duration.

1. https://domain.com/bucket/ram/00/01x/256k/program.mpd

General

```
Request URL: https://domain.com/bucket/ram/00/01x/256k/program_init.mp4?StreamS-MSE-1.0_uuid=a78e5d3b-
c956-482b-d887-3f898db3b5ea
Request Method: GET
Status Code: 200
Remote Address: 123.45.67.89:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 473
content-type: application/vnd.apple.mpegurl
date: Tue, 04 Oct 2022 23:20:05 GMT
etag: "7a20bcd047d8d81:0"
last-modified: Tue, 04 Oct 2022 23:19:56 GMT
provider: $tream$ HiFi Radio
server: StreamServer HLS
```

Request Headers

```
DNT: 1
Referer: https://la2.indexcom.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/105.0.0.0 Safari/537.36
```

2. https://domain.com/bucket/ram/00/01x/256k/program_init.mp4

General

```
Request URL: https://domain.com/bucket/ram/00/01x/256k/program_init.mp4?StreamS-MSE-1.0_uuid=a78e5d3b-
c956-482b-d887-3f898db3b5ea
Request Method: GET
Status Code: 200
Remote Address: 123.45.67.89:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 662
content-type: video/mp4
date: Tue, 04 Oct 2022 23:20:05 GMT
etag: "a1a3183dc2cbd81:0"
last-modified: Mon, 19 Sep 2022 00:53:31 GMT
provider: $tream$ HiFi Radio
server: StreamServer HLS
```

Request Headers

```
DNT: 1
Referer: https://la2.indexcom.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/105.0.0.0 Safari/537.36
```

3. https://db2.indexcom.com/bucket/ram/00/01x/256k/program-20221004T231625Z.m4s?

General

```
Request URL: https://db2.indexcom.com/bucket/ram/00/01x/256k/program-20221004T231625Z.m4s?StreamS-MSE-
1.0_uuid=a78e5d3b-c956-482b-d887-3f898db3b5ea
Request Method: GET
Status Code: 200
Remote Address: 24.199.52.152:443
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers

```
accept-ranges: bytes
access-control-allow-headers: Range
access-control-allow-methods: HEAD,GET
access-control-allow-origin: *
access-control-request-headers: Authorization,Content-Type
cache-control: no-cache,no-store
content-length: 322792
content-type: video/mp4
date: Tue, 04 Oct 2022 23:20:05 GMT
etag: "8381f8b247d8d81:0"
last-modified: Tue, 04 Oct 2022 23:19:06 GMT
provider: $tream$ HiFi Radio
```

```
server: StreamServer HLS
```

Request Headers

```
DNT: 1
Referer: https://la2.indexcom.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/105.0.0.0 Safari/537.36
```

DASH Server – Player Request – SSL HTTP/2

User-Agent: cURL

```
root@debian:~# curl -v https://domain.com/00/dash/program.mpd
*   Trying 127.0.0.1:443...
* Connected to domain.com (127.0.0.1) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use h2
* Server certificate:
*  subject: CN=domain.com
*  start date: Jul 12 00:00:00 2023 GMT
*  expire date: Jun 12 23:59:59 2024 GMT
*  subjectAltName: host "domain.com" matched cert's "domain.com"
*  issuer: C=FR; ST=Paris; L=Paris; O=Gandi; CN=Gandi Standard SSL CA 2
*  SSL certificate verify ok.
* Using HTTP2, server supports multi-use
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* Using Stream ID: 1 (easy handle 0x559a20d55fe0)
> GET /00/dash/program.mpd HTTP/2
> Host: domain.com
> user-agent: curl/7.74.0
> accept: */*
>
* Connection state changed (MAX_CONCURRENT_STREAMS == 100)!
< HTTP/2 200
< cache-control: no-cache,no-store
< content-type: application/dash+xml
< last-modified: Fri, 11 Aug 2023 06:23:45 GMT
< accept-ranges: bytes
< etag: "b94e2611cccd91:0"
< provider: StreamS HiFi
< access-control-request-headers: Authorization,Content-Type
< access-control-allow-methods: HEAD,GET
< access-control-allow-origin: *
< server: StreamS/2.0
< access-control-allow-headers: Range
< date: Fri, 11 Aug 2023 06:31:34 GMT
< content-length: 1310
<
<?xml version="1.0" encoding="utf-8"?>
```

```
<!--ENCODER:StreamS Live HTTP Encoder v3.2.1.183-->
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns="urn:mpeg:dash:schema:mpd:2011"
   xmlns:dvb="urn:dvb:dash:dash-extensions:2014-1"
   xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-DASH_schema_files/DASH-MPD.xsd"
   profiles="urn:dvb:dash:profile:dvb-dash:2014,urn:dvb:dash:profile:dvb-dash:isoff-ext-live:2014"
   type="dynamic"
   availabilityStartTime="2023-08-11T06:23:05Z" publishTime="2023-08-11T06:22:55Z"
   minimumUpdatePeriod="PT10S" timeShiftBufferDepth="PT1M0S"
   maxSegmentDuration="PT10S" minBufferTime="PT5S">

        <UTCTiming schemeIdUri="urn:mpeg:dash:utc:http-iso:2014" value="http://time.akamai.com/?iso"/>

        <BaseURL>./</BaseURL>

        <Period start="PT0S">

                <AdaptationSet contentType="audio" mimeType="video/mp4"
                   bitstreamSwitching="true" segmentAlignment="true"
                   minBandwidth="128000" maxBandwidth="128000"
                   audioSamplingRate="48000" startWithSAP="1">

                        <SegmentTemplate timescale="48000"
                           initialization="$RepresentationID$/program_init.mp4"
                           media="$RepresentationID$/program-$Number$.m4s"
                           startNumber="1" duration="479232"/>
                        <Representation id="128k" bandwidth="128000" codecs="mp4a.40.2" />

                </AdaptationSet>

        </Period>
</MPD>
* Connection #0 to host domain.com left intact
root@debian:~#
```

Until it is disconnected from the stream, the player client will continue to retrieve .m4s segments based on the encoder-specified segment size.

Note that there is no stream PAD or Now-Playing metadata exposed at this level. Metadata is contained in the .m4s segments and must be parsed out using HTML5-MSE JavaScript ISO BMFF Box and id3 frame tools. It is located in ISO BMFF emsg box as id3v2.4 frames and is UTF8mb4 encoded.

Audio is then passed to the operating system decoders using HTML5-MSE JavaScript.

The player client should generate a unique UUID/GUID for each session, which should be included in the HTTP GET Requests as a query string. This will help create a server log that can be parsed for accurate analytics and performance reporting. *Failure to do this can cause erroneous performance reporting, producing incorrect monetization calculations.*

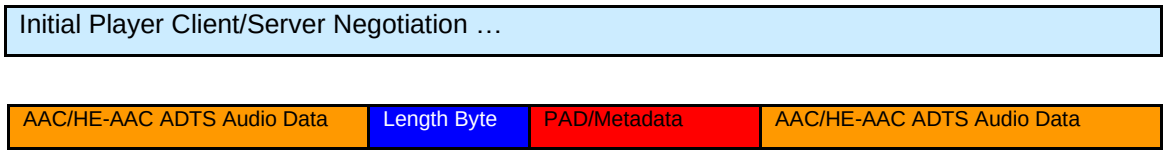
The HTML5-MSE JavaScript to accomplish this is beyond the scope of this document.

A.3 SHOUTcast v1

Metadata handling in HLS and MPEG/DASH is fundamentally different from legacy ICY stream metadata (as used in SHOUTcast v1), which has several limitations. Furthermore, the simple HTML5 <audio> tag, commonly used by many providers, does not support ICY metadata because there is no simple way to tell the ICY streaming server to send the metadata, which is not realtime anyway. Time-based query methods are therefore commonly used. These methods cannot guarantee that the program and metadata have identical latency and will be synchronized at the player. This puts time-based PAD metadata even more out of sync with the audio, and it often displays even when there is no audio playing. Hence, time-based query methods should be avoided.

SHOUTcast v1 is the simplest ICY streaming protocol. It has the fewest number of features and options, making it challenging for content providers to administer and use for large-scale deployment requiring advanced features such as authentication, mountpoints, and extended metadata. Some of the metadata limitations can be overcome by using the extensible extended ID3 PAD/metadata methods described in this document.

Two sources supply the SHOUTcast Player Client PAD/metadata. The initial Player Client/Server negotiation returns the Static PAD/metadata, and the Dynamic PAD/metadata is sent in-line with the audio data. Note that the metadata is not ID3 format.



Initial Player Client/Server negotiation:

```
GET / HTTP/1.0\r\n
Host: 12.34.56.78\r\n
User-Agent: Fictitious Audio Player 1.0 (os= Windows 10 21H1 – 10.0.19043.985)\r\n
Accept: */*\r\n
Icy-MetaData:1\r\n
Connection: close\r\n
\r\n

ICY 200 OK\r\n
icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>\r\n
icy-notice2:SHOUTcast Distributed Network Audio Server/FreeBSD v1.9.5<BR>\r\n
icy-name:Stream Name\r\n
icy-genre:Various\r\n
icy-url:http://www.domain.com\r\n
content-type:audio/aacp\r\n
icy-pub:1\r\n
icy-metaint:8192\r\n
icy-br:32\r\n
\r\n
```

The SHOUTcast Distributed Network Audio Server caches the dynamic metadata, allowing it to be available to the Player Client immediately upon connection. The six steps in the Player Client/Server interaction are as follows:

1. **Request Static PAD/metadata**

The initial Player Client/Server negotiation request will return the Static PAD/metadata.

Stream Name

```
icy-name:Stream Name
```

Website

```
icy-url: http://www.domain.com
```

The Website field is optional and is usually used to indicate the associated stream website.

Audio Codec

```
content-type:audio/aacp or content-type:audio/aac
```

Both content types are valid and should be supported to allow stream compatibility among various streaming encoders and content providers.

Audio Bit Rate

```
icy-br:32
```

Audio Sample Rate—This must be obtained by parsing an ADTS audio frame.

2. Request Dynamic PAD/metadata

Include a request to the streaming server to return PAD/metadata in the Initial Client/Server negotiation.

```
Icy-MetaData:1\r\n
```

If PAD/metadata is requested with the Icy-Metadata:1\r\n entry, the Player client must parse it. Otherwise the audio will glitch whenever the PAD/metadata streams, as there is no way for the audio decoder to know what is audio or PAD/metadata.

3. Get PAD/metadata interval

The streaming server will return the PAD/metadata interval.

```
icy-metaint:8192\r\n
```

This is the number of ADTS audio data bytes that are between metadata blocks.

This is not the same default value as Icecast2. *When writing a player to support both ICY SHOUTcast and Icecast2 protocols, note that the default metadata interval specified in icy-metaint is different for each server.* For example, using the default value for SHOUTcast will cause the default value for Icecast2 streams to glitch. It is important *not* to assume that the stream is using these default values, as they can be changed on the streaming server. This is a common bug in Player/Client software.

4. Get PAD/metadata from the stream

Read the streaming data, keeping a byte count. When the number of bytes equals the metadata interval, a metadata block is complete. The first part of the metadata block is a single byte length specifier, immediately followed by the actual PAD/metadata.

Length Specifier Byte = Metadata Length / 16. Multiply this byte by 16 to get the actual metadata length. Max Byte Size = 255 = Metadata Max Length = 4080 Characters

- Read the number of metadata bytes, which will be the number of bytes in the string containing the PAD/metadata.
- Restart the byte count, and repeat.

When counting the interval, do not count the bytes in the metadata length field or the actual metadata. Only count the audio ADTS data.

Note that the PAD/metadata length is set to “0” most of the time, meaning there is no metadata. The metadata is sent immediately after connecting to the server and whenever there is an event change that causes the streaming encoder to send PAD/metadata.

5. Parse PAD/metadata

The in-line PAD/metadata will be formatted as:

```
StreamTitle='Artist Goes Here–Title Goes Here';StreamUrl='http://www.domain.com/image.jpg'
```

To separate Artist and Title fields (which are concatenated in the metadata) on two separate lines of the display, it may be desirable to parse for the "space-hyphen-space" delimiter between them. However for displays having a limited number of lines, it may instead be necessary to display all this metadata on one scrolling line.

The StreamUrl field should not be displayed. It is used to reference image graphics or extended PAD/metadata XML, which should be parsed and displayed in Players that are capable. If this field is URL encoded, URL decoding is necessary to allow parsing of the resulting XML.

6. Display PAD/metadata

Write the PAD/metadata to your display software or device in your desired format. The display options are almost limitless in a software player. Hardware players can be a challenge. There are some design concepts at the end of this section.

- Hls/dash—in-band metadata
- Parse m3u8 playlist file—master/variant
- Parse hls segments
- Parse iso bmff
- Parse emsg (for cmaf fmp4)
- Parse id3 (segments may contain more than one id3 frame, depending upon content requirements.)
- Display visible metadata
- Process non-displayable metadata
- Play audio/video through html5 mse

ICY Server – Icecast with SHOUTcast Source – Player Request – SSLHTTP/1.1
User-Agent: Browser – HTML5 Audio Element Request

Note: There is *no* icy-metaint: The Server Header is requested and returned from the ICY Server.

The HTML5 Audio Element does not directly support metadata, making it difficult to display metadata from ICY streams

Many developers and content providers use this method to play ICY streams because it is simple and extremely easy to implement. The problem is lack of metadata support, often leading to use of poor performance, proprietary out-of-band metadata.

The HTML5 <audio> Element by itself without additional JavaScript and/or MSE is designed for playing audio files.

By the time JavaScript is added to render metadata, the complexity approaches that of HLS, however, the metadata will always be asynchronous with several other limitations, so it makes no sense to use ICY for professional streaming now, unless simplicity is the goal.

```
GENERAL
Request URL: https://ice0.domain.com/stream-128-aac
Request Method: GET
Status Code: 200 OK
Remote Address: 198.24.44.211:443
Referrer Policy: strict-origin-when-cross-origin

RESPONSE HEADERS
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Origin, Accept, X-Requested-With, Content-Type, Icy-MetaData
```

```
Access-Control-Allow-Methods: GET, OPTIONS, SOURCE, PUT, HEAD, STATS
Access-Control-Allow-Origin: https://db2.indexcom.com
Cache-Control: no-cache, no-store
Connection: Close
Content-Type: audio/aac
Date: Wed, 05 Apr 2023 05:36:05 GMT
Expires: Mon, 26 Jul 1997 05:00:00 GMT
icy-br: 128
icy-genre: Ambient Chill
icy-name: Stream Name
icy-notice1: <BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
icy-notice2: SHOUTcast Distributed Network Audio Server/Linux v1.9.5<BR>
icy-pub: 0
icy-url: http://domain.com
Server: Icecast 2.4.0-kh15

REQUEST HEADERS
Accept: */*
Accept-Encoding: identity;q=1, *;q=0
Accept-Language: en-US,en;q=0.9
Connection: keep-alive
DNT: 1
Host: ice0.domain.com
Origin: https://db2.indexcom.com
Range: bytes=0-
Referer: https://db2.indexcom.com/
sec-ch-ua: "Google Chrome";v="111", "Not(A:Brand";v="8", "Chromium";v="111"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Sec-Fetch-Dest: audio
Sec-Fetch-Mode: cors
Sec-Fetch-Site: cross-site
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/111.0.0.0 Safari/537.36
```

ICY Server – Icecast with SHOUTcast Source – Player Request – SSL-HTTP/1.1
User-Agent: cURL WITHOUT ICY-Metadata: One Request

Note: The ICY Server does *not* accept and return an icy-metaint: Server Request and Header.

```
root@debian:~# curl -s -v https://ice0.domain.com/stream-128-aac
*   Trying 198.24.44.211:443...
* Connected to ice0.domain.com (198.24.44.211) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server did not agree to a protocol
* Server certificate:
*  subject: CN=*.domain.com
*  start date: Nov  6 00:00:00 2022 GMT
*  expire date: Dec  6 23:59:59 2023 GMT
*  subjectAltName: host "ice0.domain.com" matched cert's "*.domain.com"
*  issuer: C=GB; ST=Greater Manchester; L=Salford; O=Sectigo Limited; CN=Sectigo RSA Domain Validation
Secure Server CA
*  SSL certificate verify ok.
> GET /stream-128-aac HTTP/1.1
> Host: ice0.domain.com
> User-Agent: curl/7.74.0
> Accept: */*
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Content-Type: audio/aacp
< Date: Wed, 26 Apr 2023 07:39:19 GMT
< icy-br:128
< icy-genre:Ambient Chill
< icy-name:Stream Name
< icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
< icy-notice2:SHOUTcast Distributed Network Audio Server/Linux v1.9.5<BR>
< icy-pub:0
< icy-url:http://domain.com
< Server: Icecast 2.4.0-kh15
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Connection: Close
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Origin, Accept, X-Requested-With, Content-Type, Icy-MetaData
< Access-Control-Allow-Methods: GET, OPTIONS, SOURCE, PUT, HEAD, STATS
```

ICY Server – Icecast with SHOUTcast Source – Player Request – SSL-HTTP/1.1 User-Agent: cURL *With* ICY-Metadata: Metadata Request

The ICY Server accepts and returns an icy-metaint: Client Request and Server Header.

```
root@debian:~# curl -s -H "Icy-MetaData: 1" -v https://ice0.domain.com/stream-128-aac
*   Trying 198.24.44.211:443...
* Connected to ice0.domain.com (12.34.56.78) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server did not agree to a protocol
* Server certificate:
*  subject: CN=*.domain.com
*  start date: Nov  6 00:00:00 2022 GMT
*  expire date: Dec  6 23:59:59 2023 GMT
*  subjectAltName: host "ice0.domain.com" matched cert's "*.domain.com"
*  issuer: C=GB; ST=Greater Manchester; L=Salford; O=Sectigo Limited; CN=Sectigo RSA Domain Validation
Secure Server CA
*  SSL certificate verify ok.
> GET /stream-128-aac HTTP/1.1
> Host: ice0.domain.com
> User-Agent: curl/7.74.0
> Accept: */*
> Icy-MetaData: 1
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Content-Type: audio/aac
< Date: Wed, 26 Apr 2023 04:45:48 GMT
< icy-br:128
< icy-genre:Ambient Chill
< icy-name:Stream Name
< icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
< icy-notice2:SHOUTcast Distributed Network Audio Server/Linux v1.9.5<BR>
< icy-pub:0
< icy-url:http://domain.com
< Server: Icecast 2.4.0-kh15
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Connection: Close
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Origin, Accept, X-Requested-With, Content-Type, Icy-MetaData
< Access-Control-Allow-Methods: GET, OPTIONS, SOURCE, PUT, HEAD, STATS
< icy-metaint:16000
```

ICY Server – Icecast with SHOUTcast Source – Player Request – HTTP/1.1 User-Agent: cURL *Without* ICY-Metadata: Metadata Request

The ICY Server does *not* accept and return an icy-metaint: Server Request and Header.

Things get much simpler without SSL but there is a security penalty: the encoder passwords are very easy to discover. Therefore, running ICY streams without SSL is not recommended

```
root@debian:~# curl -s -v http://ice0.domain.com/stream-128-aac
*   Trying 12.34.56.78:80...
* Connected to ice0.domain.com (12.34.56.78) port 80 (#0)
> GET /stream-128-aac HTTP/1.1
> Host: ice0.domain.com
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Content-Type: audio/aacp
< Date: Wed, 26 Apr 2023 07:48:24 GMT
< icy-br:128
< icy-genre:Ambient Chill
< icy-name:Stream Name
< icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
< icy-notice2:SHOUTcast Distributed Network Audio Server/Linux v1.9.5<BR>
< icy-pub:0
< icy-url:http://domain.com
< Server: Icecast 2.4.0-kh15
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Connection: Close
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Origin, Accept, X-Requested-With, Content-Type, Icy-MetaData
< Access-Control-Allow-Methods: GET, OPTIONS, SOURCE, PUT, HEAD, STATS
```

ICY Server – Icecast with SHOUTcast Source – Player Request – HTTP/1.1 User-Agent: cURL *With* ICY-Metadata: Metadata Request

The ICY Server accepts and returns an icy-metaint: Client Request and Server Header.

```
root@debian:~# curl -s -H "Icy-MetaData: 1" -v http://ice0.domain.com/stream-128-aac
*   Trying 12.34.56.78:80...
* Connected to ice0.domain.com (12.34.56.78) port 80 (#0)
> GET /stream-128-aac HTTP/1.1
> Host: ice0.domain.com
> User-Agent: curl/7.74.0
> Accept: */*
> Icy-MetaData: 1
>
* Mark bundle as not supporting multiuse
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Content-Type: audio/aacp
< Date: Wed, 26 Apr 2023 08:27:29 GMT
< icy-br:128
< icy-genre:Ambient Chill
< icy-name:Stream Name
< icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
< icy-notice2:SHOUTcast Distributed Network Audio Server/Linux v1.9.5<BR>
< icy-pub:0
< icy-url:http://domain.com
< Server: Icecast 2.4.0-kh15
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Connection: Close
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Origin, Accept, X-Requested-With, Content-Type, Icy-MetaData
```

```
< Access-Control-Allow-Methods: GET, OPTIONS, SOURCE, PUT, HEAD, STATS
< icy-metaint:16000
```

A.4 Icecast2

Icecast2 uses a simple ICY streaming protocol that addresses many of the SHOUTcast server limitations. It is therefore a better choice to stream when using the ICY format. It adds advanced features such as authentication and mountpoints, which allow one instance of the server to handle multiple stream sources. This makes it easier for larger content providers to administer and to use for large-scale deployment. However like SHOUTcast, Icecast2 has limited advanced metadata capabilities. Some of the metadata limitations can be overcome by using the extensible extended PAD/metadata methods described in this documentation.

Two sources supply the Icecast2 Player Client PAD/metadata. The initial Player Client/Server negotiation returns the Static PAD/Metadata, and the Dynamic PAD/metadata is sent in-line with the audio data. The metadata is not ID3 format.

Initial Player Client/Server Negotiation…

Icecast/SHOUTcast bitstream:

Initial Player Client/Server negotiation:

```
GET /aac HTTP/1.0\r\n
Host: 12.34.56.78\r\n
User-Agent: Fictitious Audio Player 1.0 (os= Windows 10 21H1 – 10.0.19043.985)\r\n
Accept: */*\r\n
Icy-MetaData:1\r\n
Connection: close\r\n
\r\n

HTTP/1.0 200 OK\r\n
Content-Type: audio/aac\r\n
icy-br:32\r\n
ice-audio-info: ice-bitrate=32;ice-samplerate=32000;ice-channels=2\r\n
icy-description:Icecast Audio Stream\r\n
icy-genre:Various\r\n
icy-name:Stream Name\r\n
icy-pub:0\r\n
icy-url:http://www.domain.com\r\n
Server: Icecast 2.3.1\r\n
icy-metaint:16000\r\n
\r\n
```

The Icecast2 Server caches the dynamic metadata, allowing it to be available to the Player Client immediately upon connect. The steps in the Player Client/Server interaction are as follows:

1. **Request Static PAD/metadata**

The initial Player Client/Server negotiation request will return the Static PAD/metadata.

Stream Name

icy-name:Stream Name

Stream Description

```
icy-description:Stream Description
```

Website

```
icy-url: http://www.domain.com
```

The Website field is optional, and is usually used to indicate the associated stream website.

Audio Codec

```
Content-Type:audio/aac or content-type:audio/aacp
```

Both content types are valid and should be supported to allow stream compatibility among various streaming encoders and content providers

Audio Sample Rate

Audio Bit Rate

Audio Channel Number (Mono-1/Stereo-2/Surround-6)

```
ice-audio-info: ice-bitrate=32;ice-samplerate=32000;ice-channels=2
```

Note that unlike SHOUTcast, it is not necessary to parse ADTS audio frames to obtain the Audio Sample Rate.

Server

```
Server: Icecast 2.3.3
```

2. Request Dynamic PAD/metadata

Include a request to the streaming server to return PAD/metadata in the Initial Client/Server negotiation.

```
Icy-MetaData:1\r\n
```

If PAD/metadata is requested with the Icy-Metadata:1\r\n entry, the Player client must parse it. Otherwise the audio will glitch whenever the PAD/metadata streams because there is no way for the audio decoder to distinguish the audio from the PAD/metadata.

3. Get the PAD/metadata interval.

The streaming server will return the PAD/metadata interval.

```
icy-metaint:16000\r\n
```

This is the number of ADTS audio data bytes that are between metadata blocks.

Note that this is not the same default value as SHOUTcast.

When writing a player to support both ICY SHOUTcast and Icecast2 protocols, note that the default metadata interval specified in icy-metaint is different for each server. For example, using the default value for SHOUTcast will cause the default value for Icecast2 streams to glitch. It is important *not* to assume that the stream is using these default values because they can be changed on the streaming server. This is a very common bug in Player/Client software.

4. Get PAD/metadata from the stream.

Read the streaming data, keeping a byte count. When the number of bytes equals the metadata interval, a metadata block is complete. The first part of the metadata block is a single byte length specifier, immediately followed by the actual PAD/metadata.

Length Specifier Byte = Metadata Length / 16. Multiply this byte by 16 to get the actual metadata length. Max Byte Size = 255 = Metadata Max Length = 4080 Characters

- Read the number of metadata bytes, which will be the string containing the PAD/metadata.
- Restart byte count, and repeat.

Be certain not to count the bytes in the metadata length field or the actual metadata, when counting the interval. Only the audio ADTS data should be counted.

Note that the PAD/metadata length is set to “0” most of the time, meaning there is no metadata. The metadata is sent immediately after connecting to the server and whenever there is an event change that causes the streaming encoder to send PAD/metadata.

5. Parse PAD/metadata

The in-line PAD/metadata will be formatted as:

```
StreamTitle='Artist Goes Here—Title Goes Here';StreamUrl='http://www.domain.com/image.jpg'
```

To display Artist and Title on two separate lines of the display, it might be desirable to parse for the "space-hyphen-space" between them. However for displays having a limited number of lines, it may instead be necessary to display all this metadata on one scrolling line.

The StreamUrl field should not be displayed. It is used to reference image graphics, or extended PAD/metadata XML which should be parsed and displayed in players that are capable. If this field is URL encoded, URL decoding is necessary to allow parsing of the resulting XML.

Note that the StreamUrl= field is only currently available in the Icecast2 Karl Heyes Build 2.3.2-kh-5 or higher.

The Karl Heyes Icecast2 Streaming Server builds are available here:

https://github.com/karlheyes/icecast-kh

6. Display PAD/metadata

Write the PAD/metadata to your display software or device in your desired format. The display options are almost limitless in a software player. Software players offer versatile display options, as shown in Figure 4 of this NRSC Guideline.

ICY Server – Icecast – Player Request – HTTP/1.1
User-Agent: cURL WITH ICY-Metadata: Request

Note: The ICY Server returns an “Icy-Metaint: (in bytes)” Client Request and Server Header.

Notice the non-standard Port 9090 this stream uses. This is typical of ICY streams running on resource-bound servers. Ports 80 and 443, which should be used, are probably in use for a webserver on the same system. HLS does not suffer from this limitation, as live streams and static web content can be served from the same server, using common HTTP 80 and HTTPS 443 Ports. Using non-standard ports for streaming creates firewall content and network security problems.

```
root@debian:~# curl -s -H "Icy-MetaData: 1" -v http://live.domain.com:9090/Stream
*   Trying 12.34.56.78:9090...
* Connected to live.domain.com (12.34.56.78) port 9090 (#0)
> GET /Stream HTTP/1.1
```

```
> Host: live.domain.com:9090
> User-Agent: curl/7.74.0
> Accept: */*
> Icy-MetaData: 1
>
* Mark bundle as not supporting multiuse
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Server: Icecast 2.4.4
< Connection: Close
< Date: Mon, 15 May 2023 02:52:11 GMT
< Content-Type: audio/aac
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Pragma: no-cache
< icy-br:192
< icy-description:Online radio station
< icy-genre:Adult Contemporary Cafe Decades
< icy-name:Stream Name
< icy-pub:1
< icy-url:https://www.domain.com
< icy-metaint:16000
<
* Failure writing output to destination
* Closing connection 0
root@debian:~#
```

ICY Server – Icecast – Player Request – SSL HTTP/1.1
User-Agent: cURL WITH ICY-Metadata: Request

The ICY Server returns an icy-metaint: (in bytes) Client Request and Server Header.

This stream uses the non-standard Port 8443. This is typical of ICY streams running on resource-bound servers. Ports 80 and 443, which should be used, are probably in use for a webserver on the same system. HLS does not suffer from this limitation, as live streams and static web content can be served from the same server using common HTTP 80 and HTTPS 443 Ports. Using non-standard ports for streaming is a firewall content and network security problem for many.

```
root@debian:~# curl -s -H "Icy-MetaData: 1" -v https://live.domain.com:8443/Stream
*   Trying 12.34.56.78:8443...
* Connected to live.domain.com (12.34.56.78) port 8443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server did not agree to a protocol
* Server certificate:
*  subject: CN=live.domain.com
*  start date: Sep 30 00:00:00 2022 GMT
*  expire date: Oct  5 23:59:59 2023 GMT
*  subjectAltName: host "live.domain.com" matched cert's "live.domain.com"
*  issuer: C=GB; ST=Greater Manchester; L=Salford; O=Sectigo Limited; CN=Sectigo RSA Domain Validation
Secure Server CA
*  SSL certificate verify ok.
> GET /Stream HTTP/1.1
> Host: live.domain.com:8443
> User-Agent: curl/7.74.0
> Accept: */*
> Icy-MetaData: 1
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Server: Icecast 2.4.4
< Connection: Close
< Date: Mon, 15 May 2023 04:53:30 GMT
< Content-Type: audio/aac
< Cache-Control: no-cache, no-store
< Expires: Mon, 26 Jul 1997 05:00:00 GMT
< Pragma: no-cache
< icy-br:192
< icy-description:Online radio station
< icy-genre:Adult Contemporary Cafe Decades
< icy-name:Stream Name
< icy-pub:1
< icy-url:https://www.domain.com
< icy-metaint:16000
<
* Failure writing output to destination
* Closing connection 0
* TLSv1.3 (OUT), TLS alert, close notify (256):
root@debian:~#
```

NRSC Document Improvement Proposal

If in the review or use of this document a potential change appears needed for safety, health or technical reasons, please fill in the appropriate information below and email, mail or fax to:

National Radio Systems Committee
c/o Consumer Technology Association
Technology & Standards Department
1919 S. Eads St.
Arlington, VA 22202
Email: standards@CTA.tech

DOCUMENT NO.	DOCUMENT TITLE:
SUBMITTER'S NAME: COMPANY:	TEL: EMAIL:
ADDRESS:	
URGENCY OF CHANGE: _____ Immediate _____ At next revision	
PROBLEM AREA (ATTACH ADDITIONAL SHEETS IF NECESSARY): a. Clause Number and/or Drawing: b. Recommended Changes: c. Reason/Rationale for Recommendation:	
ADDITIONAL REMARKS:	
SIGNATURE:	DATE:

FOR NRSC USE ONLY

Date forwarded to NAB Tech: _____
Responsible Committee: _____
Co-chairmen: _____
Date forwarded to co-chairmen: _____

Consumer Technology Association