Title **In-advance signaling of MPEG containers content**
Author Cyril Concolato

Abstract: This document summarizes various aspects related to the signaling of MPEG containers content, reviews different technologies and recommends best practices.

1 Introduction

MPEG defines several container formats, in particular ISOBMFF and MPEG-2 TS. Files conformant to these formats may contain multiple media streams, each of which may conform to different media formats, with different profiles and levels. There are several file consumption scenarios under which the full content of the file is not available to a player but under which the player has nevertheless to take a decision to retrieve the file or not. These scenarios include progressive file download, adaptive streaming, etc. In such scenarios, the player needs to have sufficient information to determine if it has or not the capabilities of playing the entire content or only a part of the container content, and when multiple container files are provided, to enable a player to choose the most appropriate file(s) to process. The practice to send information about the container content, together with URL(s) to the content and prior to its retrieval, is called hereafter "in-advance signaling". This document reviews what the current practices are, what the challenges are and recommends best practices for MPEG container formats, in particular for the ISOBMFF.

2 Requirements

In-advance signaling of the container content is usually subject to the following requirements:

- should provide sufficient information for a player to decide if it needs and can process the file or not

The information is content dependent, and typically includes:
    - the type of the container format (e.g. ISOBMFF vs MPEG-2 TS) and required features of the container format (e.g. brands of ISOBMFF)
    - the number of media streams in the container
    - the detailed coding type of each media stream
        - This includes the media type (audio, video, subtitle, metadata), codec information (HEVC, AAC, …) and codec-specific information. It is assumed that each codec defines a way to signal minimum requirements to decode the stream, such as profile and level indications.
    - the characteristics of each media stream
        - This typically include basic description such as width and height of a video stream, number of channels and sample rate for an audio stream, language information, …
    - pre-decoding capabilities required to process each stream
        - In particular, decryption when content is protected
    - post-decoding capabilities required to correctly present each stream.
        - Post-decoding capabilities are more and more often used, in particular in video applications. When using AVC or HEVC, SEI messages are used to describe the post-processing required at the client side to properly display the video. For instance, when a video stream uses frame packing it can be

necessary to signal it in advance as a player non-capable of unpacking the frames should not decode and display the stream.

It should be noted, that most of this information (except coding type) is not dependent on the compression format used and should have (or already has) a well-defined mapping in 23001-8 CICP.

- shall be accurate

It is important that the information provided prior to the file download be correct to avoid wasted downloads. It shall describe accurately the container content. But since the signaling information is stored/sent separately from the file, there is a risk that the signaling and the content diverge.

The signaling information may be incomplete in the sense that it may not suffice for a player to decide if it needs or can play the content of the container.

In particular, when the content type or characteristics change within the file, the signaling information shall describe the entire lifetime of the content: either by providing all the different features or by providing only the common part.

- should be compatible with protocols and formats carrying such information

Early signaling information may be carried in delivery protocols (such as HTTP), in manifest formats for adaptive streaming (such as MPEG DASH). Therefore, it should respect the constraints (e.g. the encoding) of these protocols and formats. Usually, the information to be provided includes binary information, while formats and protocols are text-based. Early signaling is therefore usually encoded in a text-compatible way (e.g. base64).

- should not be too large in size

Players may have to retrieve the signaling information of multiple files before making a decision. The size of the signaling information should not therefore be too large to make this decision process efficient.

- should not interfere with manifest stream selection instructions

Many features of the manifest formats used in adaptive streaming technologies are also available in MPEG container technologies:

- the ability to describe multiple streams, possibly alternative versions of each stream
- the ability to provide stream-specific metadata enabling the selection of a media stream (language, kind, encryption, …)

Fundamentally, a manifest is a playlist. A manifest implies sequencing media content, possibly of different types and containers. In-advance signaling information should not carry information about sequencing, alternates or parallel stream selection, since such information is present in the manifest.

3 Current usages of signaling information

This section reviews some of the current usages, from the perspective of the consumer of this information, typically player devices.

3.1 Use in HTTP-based environments

In HTTP environments, headers are used to carry such information.

When using an HTTP HEAD request, a player receives the signaling information in the "Content-Type" header of the HTTP response without actually downloading the resource. A player may issue multiple HEAD requests for multiple files and then, based on the signaling information, issue a GET request for the most appropriate content.

NOTE: This scenario may not be representative of current practices. It is usually assumed that in-advance signaling information is obtained at the same time as the URL(s) to the content and that no additional round-trip (HEAD or GET) is necessary to obtain it.

Alternatively, when using the "Accept" header in a GET request, players can indicate to the server its supported features. The server can in turn respond with the most appropriate file.

NOTE: This scenario is also not a current practice and does not strictly rely on "content signaling" but rather on "capabilities signaling". It may still be relevant because one may describe its capabilities in terms of hypothetical content it could process.

In both approaches, headers are based on the MIME format (see below).

3.2 Use in HTML-based environments

In Web-based environment, in particular in HTML content, a web page can use media content coming from MPEG containers and in that case the signaling information may be embedded in the HTML content (or CSS, JavaScript, XML …) and therefore no additional round trip is required for the purpose of selecting which media content to download and play.

In particular, when multiple container files are offered, the "type" attribute on the <source> tag (used in the <video>, <audio> and <picture> tags) enables a browser to know which resource to download. The "type" attribute follows the MIME format.

In JavaScript, it is possible to ask the browser if it supports a given container format using the methods `HTMLMediaElement.canPlayType(mediaType)` and `MediaSource.isTypeSupported(mimeType)`. Both methods rely on a MIME type parameter.

Additionally, the `MediaSource.addSourceBuffer(mimeType)` API used for mapping of Adaptative Streaming protocols and formats such as MPEG DASH into browsers, also supports the use of MIME type.

Finally, the Media Capabilities API[1] provides JavaScript APIs to allow websites to make an optimal decision when picking media content for the user. The APIs will expose information about the decoding and encoding capabilities of a browser for a given format, but also output capabilities of the current device to find the best match based on the device's display. It is also in part based on the MIME type of the media container.

3.3 Use in Hardware

One use case where advance signaling could also be used is when two devices are physically connected by hardware interfaces such as HDMI and need to negotiate which content to play among a set of pieces of content.

[Editor's note: This section should contain more context, more information about the current or envisaged hardware usages]

3.4 Use in Manifest Formats

Manifest formats such as MPEG DASH declare different media resources for the selection by media players. It is therefore important for those players to distinguish between supported and non-supported resources.

In MPEG DASH, several attributes, in particular at the Representation level, enable providing in-advance signaling of the Representation features, within the manifest itself. It is also based on declaring the MIME type as well as its sub-parameters.

3.5 MIME

The Multipurpose Internet Mail Extensions (MIME) RFC 2045 defines a set of messages to exchange data and in particular syntax known as MIME Type, also known as Internet Media Type or Content Type, to describe the exchanged data. The general definition of the MIME type can and is extended for specific media data and in particular for MPEG containers by RFC6381.

3.5.1 General definition

The MIME type is a string, with a specific encoding. It is composed of a type, a subtype, and optional parameters. A reduced list of types is defined at IANA (audio, video, image, font, …) and a list of defined subtypes is maintained by IANA[2].

The MIME standard is extensible and additional sub-parameters can be defined to provide specific information for a given type/subtype. The syntax for these parameters is generic, but the allowed parameters and semantics for a given type/subtype are provided by the RFC defining the type/subtype.

3.5.2 Media specific definition

There are currently several documents defining MPEG related MIME types:

- RFC 3003 defines the audio/mpeg type
- RFC 3640 defines the video/mpeg-4 generic type
- RFC 5691 and RFC 6295 define the audio/mpeg-4 generic type
- RFC 4337 defines the MIME subtype "mp4" allowing the description of resources of type "video/mp4", "audio/mp4" and "application/mp4" (and the

[1] https://wicg.github.io/media-capabilities/
[2] https://www.iana.org/assignments/media-types/media-types.xhtml

- RFC 6381 defines two sub-parameters ("codecs" and "profiles") in particular for the above types

4 Current practices

Several options are currently used to provide in-advance signaling:

1. Use of the low-level MIME sub-parameters

This option consists in using one or more MIME sub-parameters to describe the different required capabilities (pre-decoding, decoding, and post-decoding). It is the mostly used options today because it has the advantages of enabling a progressive, detailed, compact and almost human readable signaling. The main problems are:

- The signaled information is rarely complete
- The risk of the signaling information becoming stale is high
- It is currently hard to indicate the presence/absence and parameters associated to pre-/post-decoding (encryption, SEI messages, HDCR info). Proposal exists to refine existing parameters or to define additional ones. This raises the question of the complexity of such signaling, involving many parameters.

2. Use of the "profiles" parameter

A practice used to reduce the complexity of the signaling information is to define application profiles, i.e. restricting the possible variations at different layers of an application (pre-decoding, decoding, post-decoding and rendering) and to only signal the application profile required to process a given content via the "profiles" attribute. This approach is followed by MPEG Application Formats such as CMAF.

3. Use of ISOBMFF Initialization Segments

Another practice is to transmit the entire initialization segment instead of redundant information. It has the advantages of being complete, accurate, future-proof, but the drawback of not being human readable and possibly requires transmitting more information than the other approaches. This can be illustrated by the use of Base64 encoded Initialization Segments with "data:" URLs in DASH manifest.

5 Conclusion

MPEG recommends experts to provide contributions improving this document by giving additional use cases, practices, problems and by providing guidelines on when to use the above current practices.