MPEG DASH: A Technical Deep Dive and Look at What's Next

Andy Salo
RGB Networks

Abstract

The MPEG DASH standard was ratified in December 2011 and published by the International Organization for Standards (ISO) in April 2012. This paper will review the technical aspects of the new MPEG DASH standard in detail, including: how DASH supports live, on-demand and time-shifted (NDVR) services; how the two primary video formats – ISO-base media file format (IBMFF) and MPEG-2 TS – compare and contrast; how the new standard supports digital rights management (DRM) methods; and how Media Presentation Description (MPD) XML files differ from current adaptive streaming manifests. In addition, the paper will discuss how MPEG DASH is likely to be adopted by the industry, what challenges must still be overcome, and what the implications could be for cable operators and other video service providers (VSPs).

INTRODUCTION

For much of the past decade, it was quite difficult to stream live video to a mobile device. Wide bandwidth variability, unfavorable firewall configurations and lack of network infrastructure support all created major roadblocks to live streaming. Early, more traditional streaming protocols, designed for small packet formats and managed delivery networks, were anything but firewall-friendly. Although HTTP progressive download was developed partially to get audio and video streams past firewalls, it still didn't offer true streaming capabilities.

Now, the advent of adaptive streaming over HTTP technology has changed everything, reshaping video delivery to PCs, laptops, game consoles, tablets, smartphones and other mobile devices, as well as such key home devices as Web-connected TVs and pure and hybrid IP set-top boxes (STBs). As a result, watching video online or on the go is no longer a great novelty, nor is streaming Internet-delivered content to TV screens in the home. Driven by the explosion in video-enabled devices, consumers have swiftly moved through the early-adopter phase of TV Everywhere service, reaching the point where a growing number expect any media to be available on any device over any network connection at any time. Increasingly, consumers also expect the content delivery to meet the same high quality levels they have come to know and love from traditional TV services.

Even though the emergence of the three main adaptive streaming protocols from Adobe, Apple and Microsoft over the past three and a half years has made multiscreen video a reality, significant problems still remain. Each of the three proprietary platforms is a closed system, with its own manifest format, content formats and streaming protocols. So, content creators and equipment vendors must craft several different versions of their products to serve the entire streaming video market, greatly driving up costs and restricting the market's overall development.

In an ambitious bid to solve these nagging problems, MPEG has recently adopted a new standard for multimedia streaming over the Internet. Known as MPEG Dynamic Adaptive Streaming over HTTP, or MPEG DASH, the new industry standard attempts to create a universal delivery format for streaming media by incorporating the best elements of the three main proprietary streaming solutions. In doing so, MPEG DASH aims to provide the long-sought interoperability between different network servers and different consumer electronics devices, thereby fostering a common ecosystem of content and services.

This paper will review the technical aspects of the new MPEG DASH standard in detail, including: how DASH supports live, on-demand and time-shifted (NDVR) services; how the two primary video formats (ISO-base media file format (IBMFF) and MPEG-2 TS) compare and contrast; how the standard supports DRM methods; and how Media Presentation Description (MPD) XML files differ from current adaptive streaming manifests. In addition, the paper will discuss how MPEG DASH is likely to be adopted by the industry, what challenges must still be overcome, and what the implications could be for cable operators and other video service providers (VSPs).

AN ADAPTIVE STREAMING PRIMER

As indicated previously, the delivery of streaming video and audio content to consumer electronics devices has come a long way over the past few years. Thanks to the introduction of adaptive streaming over HTTP, multimedia content can now be delivered more easily than ever before. In particular, adaptive streaming offers two critical features for video content that have made the technology the preferred choice for mobile delivery.

First, adaptive streaming over HTTP breaks down, or segments, video programs into small, easy-to-download chunks. For example, Apple's HTTP Live Streaming (HLS) protocol typically segments video content into 10-second chunks, while Microsoft's Smooth Streaming (MSS) protocol and Adobe's HTTP Dynamic Streaming (HDS) usually break video content into even smaller chunks of five seconds or less.

Second, adaptive streaming encodes the video content at multiple bitrates and resolutions, creating different chunks of different sizes. This is the truly 'adaptive' part of adaptive streaming, as the encoding enables the mobile client to choose between various bitrates and resolutions and then adapt to larger or smaller chunks automatically as network conditions keep changing.

In turn, these two key features of adaptive streaming lead to a number of benefits:

1. Video chunks can be cached by proxies and easily distributed to content delivery networks (CDNs) or HTTP servers, which are simpler and cheaper to operate than the special streaming servers required for 'older' video streaming technologies.

2. Bitrate switching allows clients to adapt dynamically to network conditions.

3. Content providers no longer have to guess which bitrates to encode for end devices.

4. The technology works well with firewalls because the streams are sent over HTTP.

5. Live and video-on-demand (VoD) workflows are almost identical. When a service provider creates a live stream, the chunks can easily be stored for later VoD delivery.

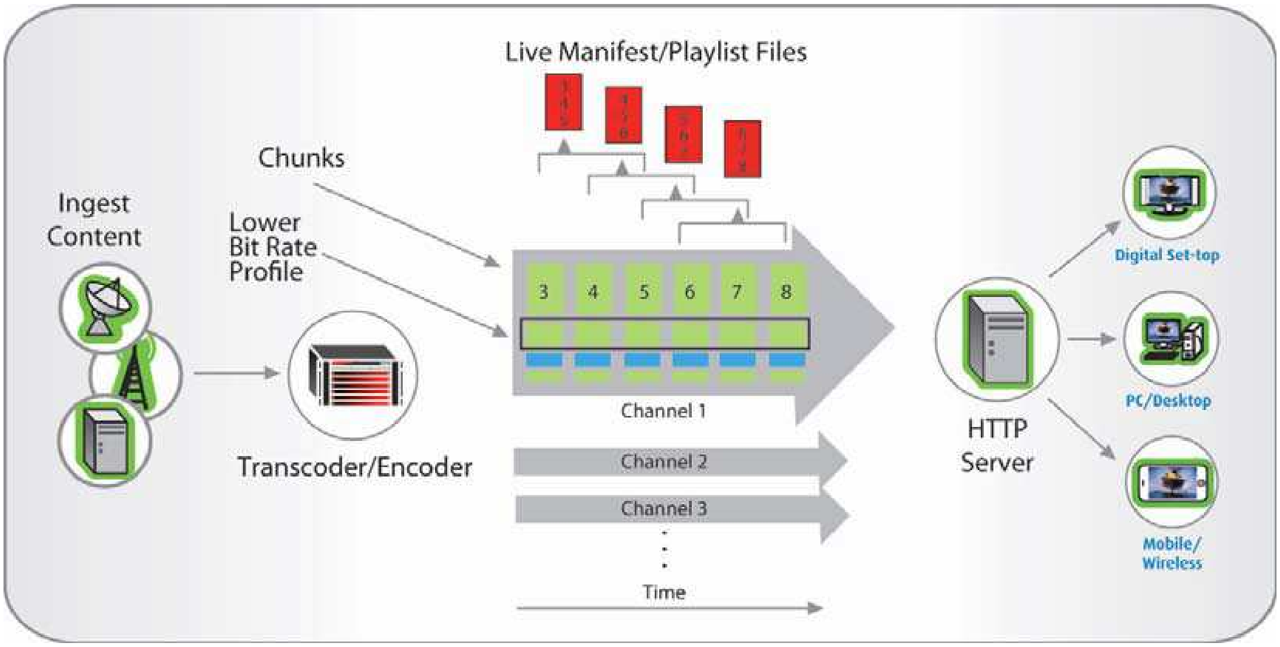

Figure 1: Content Delivery Chain for Live Adaptive Streaming

Sensing the promise of adaptive streaming technology, several major technology players have sought to carve out large shares of the rapidly growing market. Most notably, the list now includes such prominent tech companies as Adobe, Apple and Microsoft.

While the streaming of video using HTTP-delivered fragments goes back many years (and seems lost in the mists of time), Move Networks caught the attention of several media companies with its adaptive HTTP streaming technology in 2007. Move was quickly followed by Microsoft, which entered the market by releasing Smooth Streaming in October 2008 as part of its Silverlight architecture. Earlier that year, Microsoft demonstrated a prototype version of Smooth Streaming by delivering live and on-demand streaming content from such events as the Summer Olympic Games in Beijing and the Democratic National Convention in Denver.

Smooth Streaming has all of the typical characteristics of adaptive streaming. The video content is segmented into small chunks and then delivered over HTTP. Usually, multiple bitrates are encoded so that the client can choose the best video bitrate to deliver an optimal viewing experience based on network conditions.

Apple came next with HLS, originally unveiling it with the introduction of the iPhone 3.0 in mid-2009. Prior to the iPhone 3, no streaming protocols were supported natively on the iPhone, leaving developers to wonder what Apple had in mind for native streaming support. In May 2009, Apple proposed HLS as a standard to the Internet Engineering Task Force (IETF), and the draft is now in its eighth iteration.

HLS works by segmenting video streams into 10-second chunks; the chunks are stored using a standard MPEG-2 transport stream file format. The chunks may be created using several bitrates and resolutions – so-called profiles – allowing a client to switch dynamically between different profiles, depending on network conditions.

Adobe, the last of the Big Three, entered the adaptive streaming market in late 2009 with the announcement of HTTP Dynamic Streaming (HDS). Originally known as "Project Zeri," HDS was introduced in June 2010. Like MSS and HLS, HDS breaks up video content into small chunks and delivers them over HTTP. Multiple bitrates are encoded so that the client can choose the best video bitrate to deliver an optimal viewing experience based on network conditions.

HDS is closer to Microsoft Smooth Streaming than it is to Apple's HLS protocol. Primarily, this is because HDS, like MSS, uses a single aggregate file from which MPEG-4 container fragments are extracted and delivered. In contrast, HLS uses individual media chunks rather than one large aggregate file.

Feature	HLS	MSS	HDS
Multiple audio channels		☺	
Encryption		☺	☺
Closed captions / subtitling	☺	☺	
Custom VoD playlists	☺		
ability to insert ads	☺	😐	
trick modes (fast forward / rewind)		😐	
fast channel change & Stream latency		☺	☺
Client failover	☺		
Metadata	☺	☺	☺

Figure 2: Feature Comparison of Three Major Adaptive Streaming Platforms

THE DUELING STREAMING PLATFORM PROBLEM

The three major adaptive streaming protocols – MSS, HLS and HDS – have much in common. Most importantly, all three streaming platforms use HTTP streaming for their underlying delivery method, relying on standard HTTP Web servers instead of special streaming servers. They all use a combination of encoded media files and manifest files that identify the main and alternative streams and their respective URLs for the player. And their respective players all monitor either buffer status or CPU utilization and switch streams as necessary, locating the alternative streams from the URLs specified in the manifest.

The overriding problem with MSS, HLS and HDS is that these three different streaming protocols, while quite similar to each other in many ways, are different enough that they are not technically compatible. Indeed, each of the three proprietary commercial platforms is a closed system with its own type of manifest format, content formats, encryption methods and streaming protocols, making it impossible for them to work together.

Take Microsoft Smooth Streaming and Apple's HLS. Here are three key differences between the two competing platforms:

1. HLS makes use of a regularly updated "moving window" metadata index file that tells the client which chunks are available for download. Smooth Streaming uses time codes in the chunk requests so that the client doesn't have to keep downloading an index file. This leads to a second difference:

2. HLS requires a download of an index file every time a new chunk is available. That makes it desirable to run HLS with longer duration chunks, thereby minimizing the number of index file downloads. So, the recommended chunk duration with HLS is 10 seconds, while it is just two seconds with Smooth Streaming.

3. The "wire format" of the chunks is different. Although both formats use H.264 video encoding and AAC audio encoding, HLS makes use of MPEG-2 Transport Stream files, while Smooth Streaming makes use of "fragmented" ISO MPEG-4 files. The "fragmented" MP4 file is a variant in which not all the data in a regular MP4 file is included in the file. Each of these formats has some advantages and disadvantages. MPEG-2 TS files have a large installed analysis toolset and have pre-defined signaling mechanisms for things like data signals (e.g. specification of ad insertion points). But fragmented MP4 files are very flexible and can easily accommodate all kinds of data, such as decryption information, that MPEG-2 TS files don't have defined slots to carry.

Or take Adobe HDS and Apple's HLS. These two platforms have a number of key differences as well:

1. HLS makes use of a regularly updated "moving window" metadata index (manifest) file that tells the client which chunks are available for download. Adobe HDS uses sequence numbers in the chunk requests so the client doesn't have to keep downloading a manifest file.

2. In addition to the manifest, there is a bootstrap file, which in the live case gives the updated sequence numbers and is equivalent to the repeatedly downloaded HLS playlist.

3. Because HLS requires a download of a manifest file as often as every time a new chunk is available, it is desirable to run HLS with longer duration chunks, thus minimizing the number of manifest file downloads. More recent Apple client versions appear to check how many segments are in the playlist and only re-fetch the manifest when the client runs out of segments. Nevertheless, the recommended chunk duration with HLS is still 10 seconds, while it is usually just two to five seconds with Adobe HDS.

4. The "wire format" of the chunks is different. Both formats use H.264 video encoding and AAC audio encoding. But HLS makes use of MPEG-2 TS files, while Adobe HDS (and Microsoft SS) make use of "fragmented" ISO MPEG-4 files.

Due to such differences, there is no such thing as a universal delivery standard for streaming media today. Likewise, there is no universal encryption standard or player standard. Nor is there any interoperability between the devices and servers of the various

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**