Build a Simple Media-streaming Rails App

Jon Cameron
Phuong Dinh

Avalon Media System
Indiana University

Samvera Connect 2020

avalon
MEDIA SYSTEM

Setup Time

https://github.com/avalonmediasystem/connect2020-workshop/

Streaming Media Overview

Why a Streaming Server?

- Performance
- Scale
- Convenience
- Smooths away the hard parts of delivering media

What's out there?

- Local vs. cloud-hosted
- Wowza
  - Popular enterprise solution
- Amazon CloudFront
- Red5
  - Open source, often used for Flash
- HTTP Server + HLS
  - Ex: Nginx + HLS module

Protocols

- HLS – HTTP Live Streaming (Apple)
- MPEG-DASH
- RTMP
- Progressive Streaming
- Others...
- Other concerns
  - Adaptive Streaming
- Most are over HTTP, some don’t have to be

HLS

HTTP Live Streaming

- The most popular format!
- Breaks the media into chunks
- Uses M3U8 file to list and provide metadata for chunks
- Adaptive streaming: multiple bandwidth levels can be added
- Native support almost exclusively in Apple world

MPEG-DASH

(Dynamic Adaptive Streaming over HTTP)

- Dynamic Adaptive Streaming over HTTP (DASH)
- Favored and used by Google
- Developed by MPEG
- ISO Standard
- Sends media in chunks, very similar to HLS
- Codec/container agnostic
- Much more native hardware/OS support

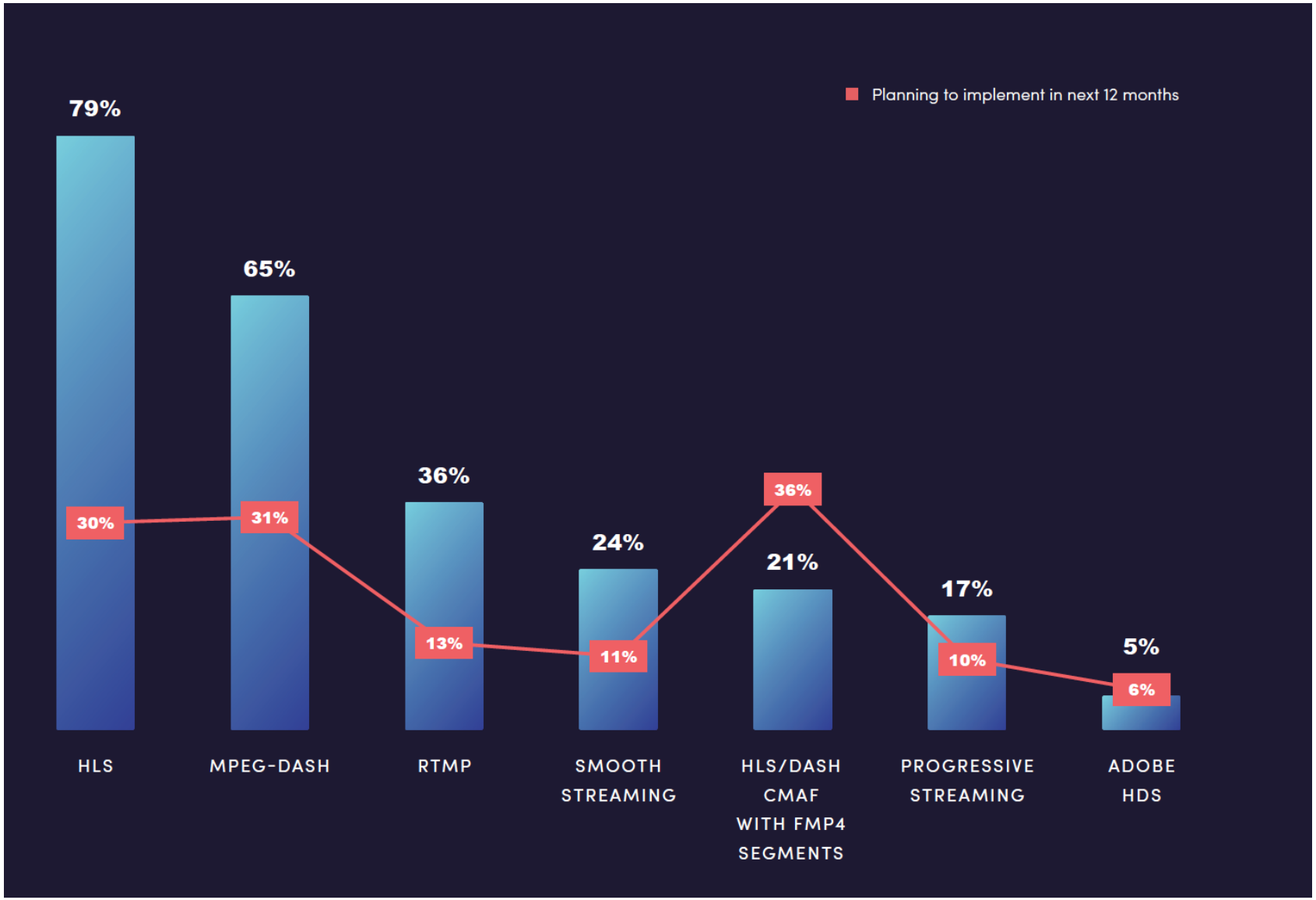
RTMP

Real-time Messaging Protocol

- It’s Flash
- Proprietary and convoluted
- Remember Flash?
- RIP

Progressive streaming

- Gets the job done
- Your media is just a resource available over HTTP
- File downloads over time
- Doesn’t scale well



2020 Bitmovin Video Developer Report

Codecs

- H.264
- VP9
- AV1
- Other friends

H.264

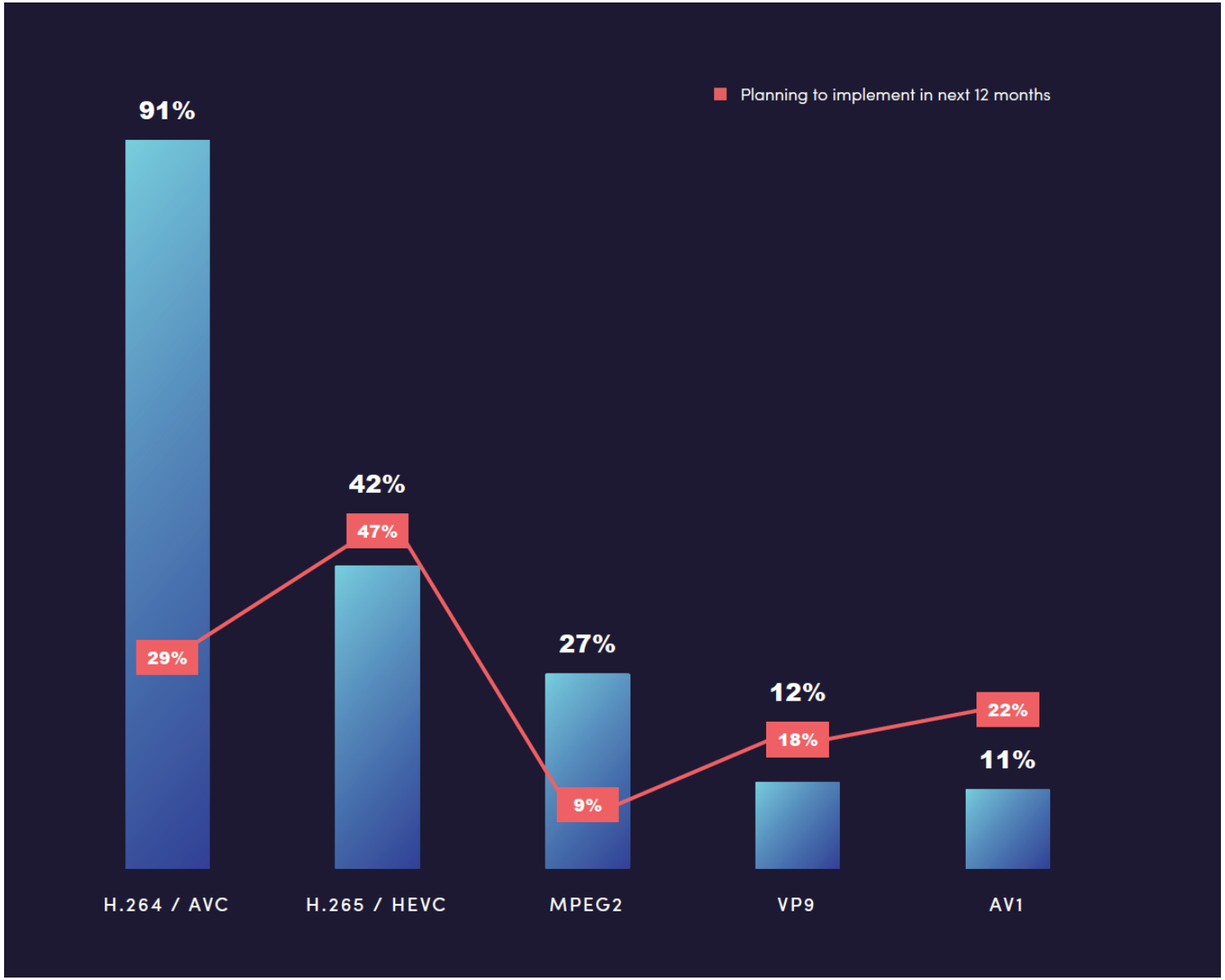
- The ubiquitous standard of the HD era
- Another hit from MPEG
- The “big dog” of video encoding
- Lots of options, good quality to file size
- Patent-encumbered
- Many encode/decode implementations

VP9

- Google’s alternative to H.265
- Better quality at lower file sizes...
- ...but less support across the board
- Native support in Google hardware

AV1

- The next big thing
- Better quality at lower sizes than all of the above
- Shepherded by the *Alliance for Open Media*
- Early implementation efforts underway
- Standards!!



2020 Bitmovin Video Developer Report

JavaScript Video Players

- Our friends
- Abstract away the inconsistencies of format and protocol
- Uses MSE (Media Source Extensions) in the browser
- dynamically construct media streams for <audio> and <video>
  - https://www.w3.org/TR/media-source/

Walkthrough of Avalon FFmpeg Presets

- Removing metadata
- Resolution
- Bitrates
- Mixdowns
- ... and more!

FFmpeg Options: Audio Encoding (high quality)

- -map_chapters -1 Remove chapter metadata
- -ac 2 Stereo Mixdown
- -ar 44100 Standard audio sample rate
- -ab 320k Bitrate: 320 kbps
- -vn Remove any existing video data
- -acodec aac Use the AAC codec
- -strict -2 Play nice with older FFmpeg versions

FFmpeg Options: Audio Encoding (medium quality)

- -map_chapters -1 Remove chapter metadata
- -ac 2 Stereo Mixdown
- -ar 44100 Standard audio sample rate
- -ab 128k Bitrate: 128 kbps
- -vn Remove any existing video data
- -acodec aac Use the AAC codec
- -strict -2 Play nice with older FFmpeg versions

FFmpeg Options: Video Encoding

- -map_chapters -1 Remove chapter metadata
- -vf Video filtering options
  - yadif=0:-1:1 Apply de-interlacing filter for interlaced media
  - scale=trunc(oh*dar/2)*2:min(ih\\,1080) Scale video down to a max of 1080px high
- -vcodec libx264 Use the x.264 library to encode h.264 video
- -preset fast Speed to compression ratio
- -profile main Use H.264 “main” profile (profile defines capability)
- -level 3.1 Profile level specifying a max data rate of 14 Mbit/s
- -pix_fmt yuv420p Set a 4:2:0 color space (compatibility!)

FFmpeg Options: Video Encoding

-b 3M	Bitrate
-maxrate 3M	Bitrate maximum
-bufsize 4M	Buffer size
-threads 0	Use as many CPU threads as are available
-force_key_frames "expr:gte(t,n_forced*2)"	Set Key Frames
-acodec aac -ab 192k -ar 44100	Audio Encoding Options
-movflags faststart	Move key information to the beginning of the file
-strict -2	Play nice with older FFmpeg versions

Authorization

- Securing those streams
- Generally token-based (can also be cookie-based)
- Authentication brokering between client and server
- Often disabled by default

> "You can use token authentication to make the stream playback URL unavailable after a certain length of time, to limit access to approved IP addresses, or apply other restrictions. Token authentication prevents playback URLs from being shared by unauthorized links or player hijacking attacks."
> ~ *Wowza Docs*

Workshop Time