W2: ENCODING 2017: CODECS & PACKAGING FOR PCS, MOBILE, & OTT/STB/SMART TVS

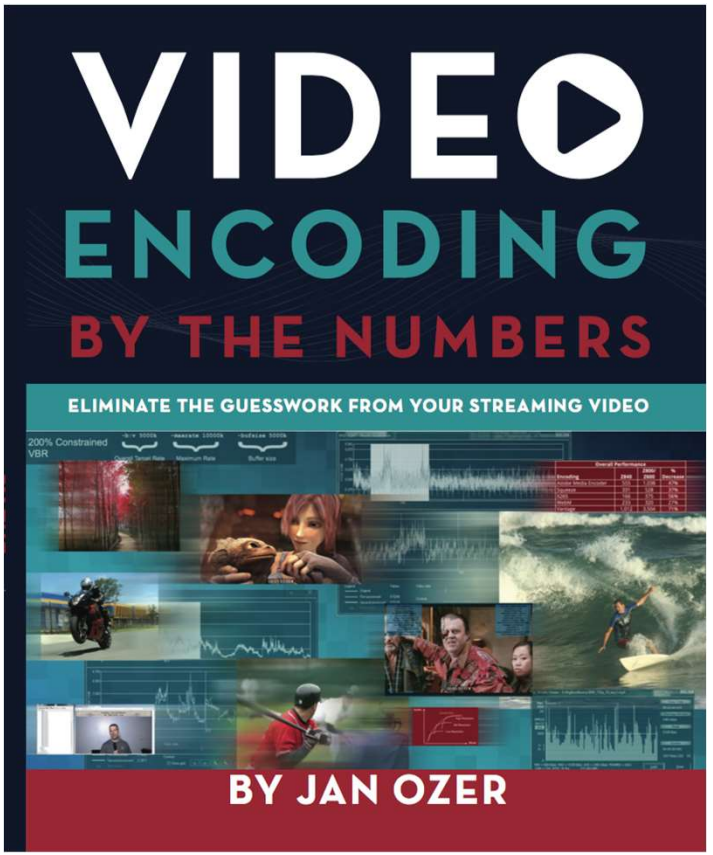
Jan Ozer
www.streaminglearningcenter.com
jozer@mindspring.com/
276-235-8542
@janozer

Agenda

- Fundamentals
- Targeting your platforms
- Producing and delivering your streams
- Configuring your streams

Shameless Plug

- All tables from this book
  - Published 12/16
  - Retail - $49.95 – print
  - PDF - $39.95
- Show special:
  - $40
  - PDF included no extra charge
  - Limited supply available (only books bought at show)



Paperback – December 28, 2016
by Jan Lee Ozer (Author)
★★★★★ 6 customer reviews

Fundamentals

- Compression related
  - Delivery paradigms
  - Bitrate control (CBR/VBR)
  - I-, B-, and P-frames
- Choosing a codec
- Codecs/container formats

Compression-Related Fundamentals

- Delivery paradigms: Single vs. Adaptive Bitrate (ABR)
- Bitrate control techniques
- I-, B- and P-frames

Adaptive Streaming

- Adaptive streaming
  - Single input file (live or VOD)
  - Encoded to multiple outputs
- Delivered adaptively based upon playback CPU and connection bandwidth
  - Technically complex, but optimizes experience across all platforms and connection types

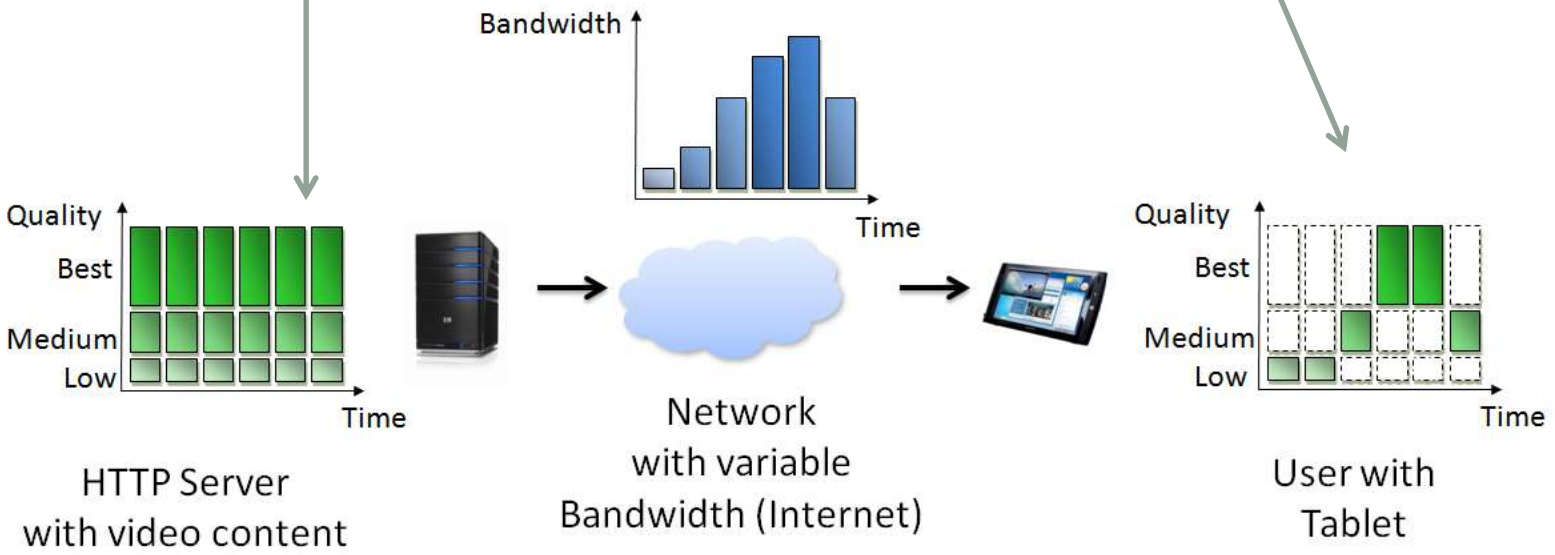


Illustration courtesy of www.bitmovin.net

Single vs. Adaptive

- Single file delivers inferior experience
  - Very simple to achieve via Gen 1 HTML5 (video tag) and support in mobile
  - This approach excludes adaptive, DRM, live, and advertising support
  - Resources:
    - Webinar: Distributing to Desktops and Mobile Devices via HTML5 (bit.ly/Ozer_MSE_EME)
    - Video tutorial: Supporting HTML5 with Flash Fallback in Squeeze 9 (bit.ly/squeeze_html5)
- Our focus is on adaptive delivery

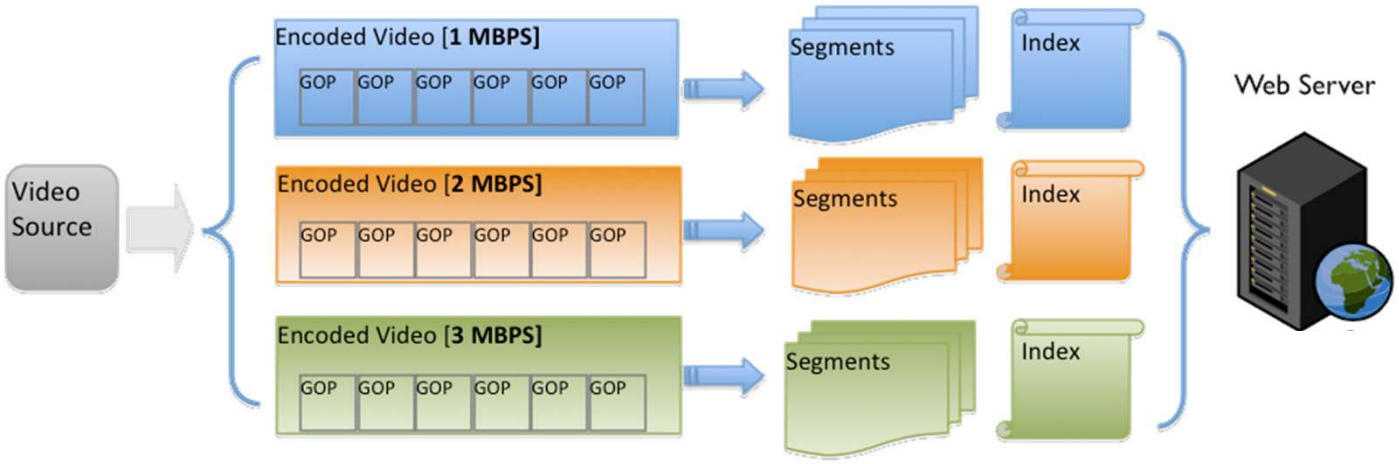
ABR Technology Overview

- Two types of systems
  - Server-based (Flash, RTMP)
    - Legacy; on the way out
  - HTTP (most new installations)
    - HTTP Live Streaming (HLS)
    - Smooth Streaming
    - HTTP-based Dynamic Streaming (HDS)
    - Dynamic Adaptive Streaming over HTTP (DASH)

Perspective

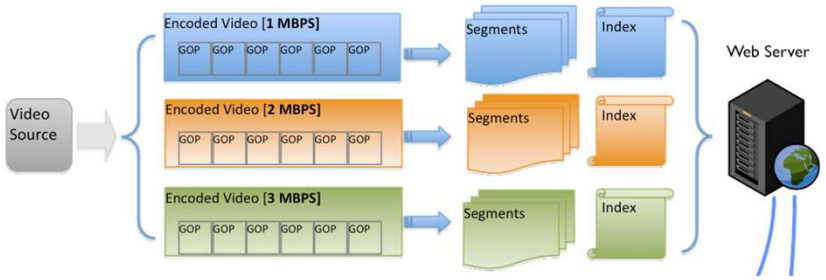
- All HTTP Technologies work the same way
  - Create chunked data files (or discrete segments within longer file)
  - Create index files (also called manifest files) with locations of chunks
  - Uploads all files to HTTP server
- Player side
  - Monitors playback buffer and (sometimes) CPU use
  - Changes streams as necessary
  - Uses index files to find the right files

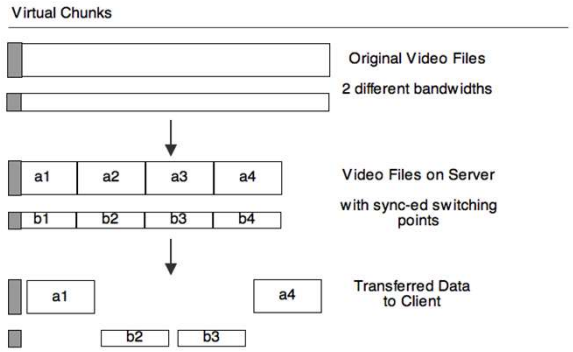
Apple HTTP Live Streaming (HLS)



- Encoder creates:
  - Chunked video files
  - Index files (M3U8) with file descriptions (rez/data rate/profile) and chunk URLs
- Uploads to HTTP web server

FILES AND BIT RANGE REQUEST

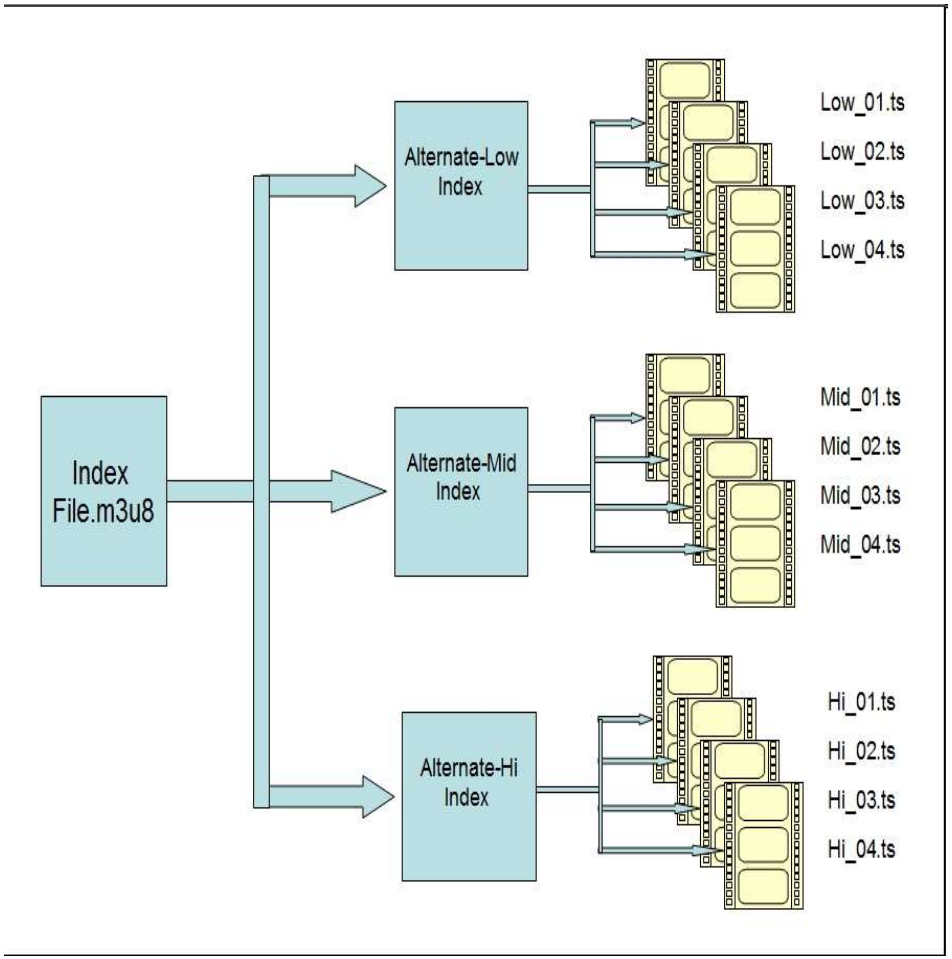




- When HTTP-based ABR started, all content files were split into multiple segments
  - Created administrative nightmare

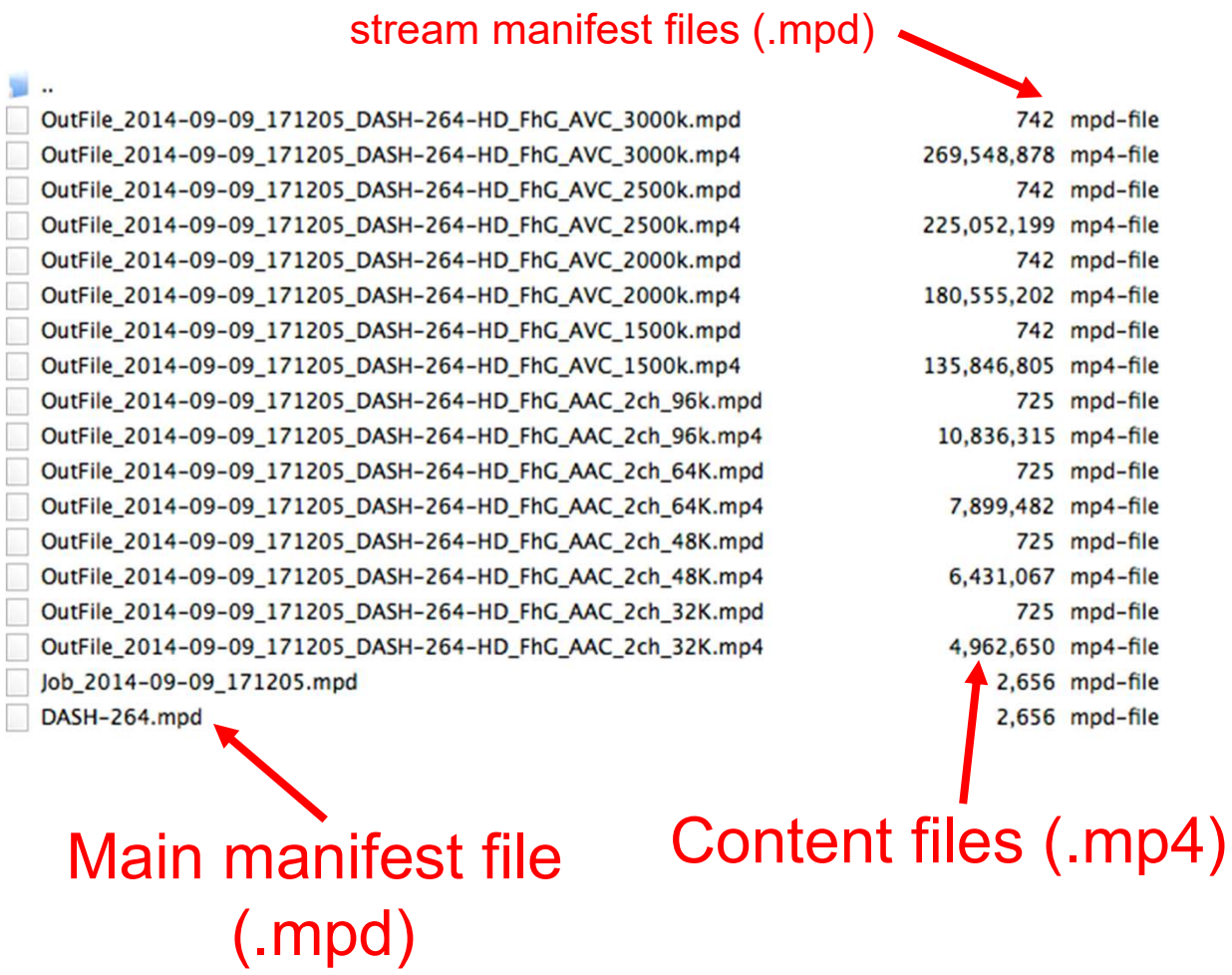
- Now all can use byte range requests from a single file
  - Upload a long single file **per layer** with data in the header that identifies the relevant segments
    - MPEG-2 ts for HLS
    - fMP4 for DASH, Smooth Streaming, HDS
- Talk about segments, mean both approaches

Manifest File Structure



- Player
  - Retrieves main index, then file index, finds first chuck
  - Monitors buffer/CPU heuristics
  - Changes streams as needed using index files to find location
    - If heuristics are good, moves to higher quality chunk
    - If heuristics are poor, moves to lower quality chunk

DASH


stream manifest files (.mpd)
..
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_3000k.mpd 742 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_3000k.mp4 269,548,878 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_2500k.mpd 742 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_2500k.mp4 225,052,199 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_2000k.mpd 742 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_2000k.mp4 180,555,202 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_1500k.mpd 742 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AVC_1500k.mp4 135,846,805 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_96k.mpd 725 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_96k.mp4 10,836,315 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_64K.mpd 725 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_64K.mp4 7,899,482 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_48K.mpd 725 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_48K.mp4 6,431,067 mp4-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_32K.mpd 725 mpd-file
OutFile_2014-09-09_171205_DASH-264-HD_FhG_AAC_2ch_32K.mp4 4,962,650 mp4-file
Job_2014-09-09_171205.mpd 2,656 mpd-file
DASH-264.mpd 2,656 mpd-file
Main manifest file (.mpd)
Content files (.mp4)

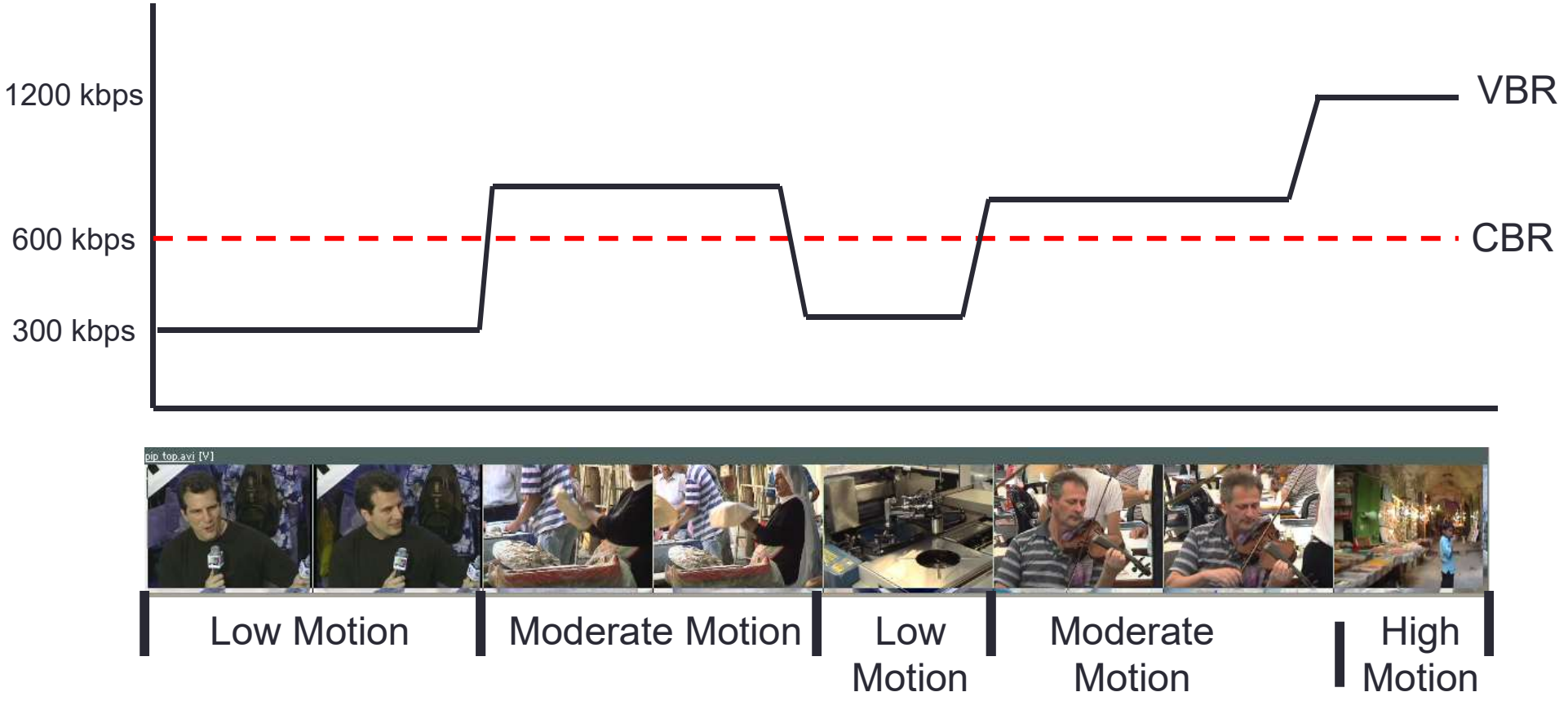
HTTP Adaptive Summary (review)

- All technologies work similarly
  - Chunked or segmented video files
  - Manifest data files
  - HTTP server
  - Player driven operation
- The big differentiating issues are:
  - Where they play
  - Whether they are a standard or proprietary
  - How much they cost (DASH=CA$H)

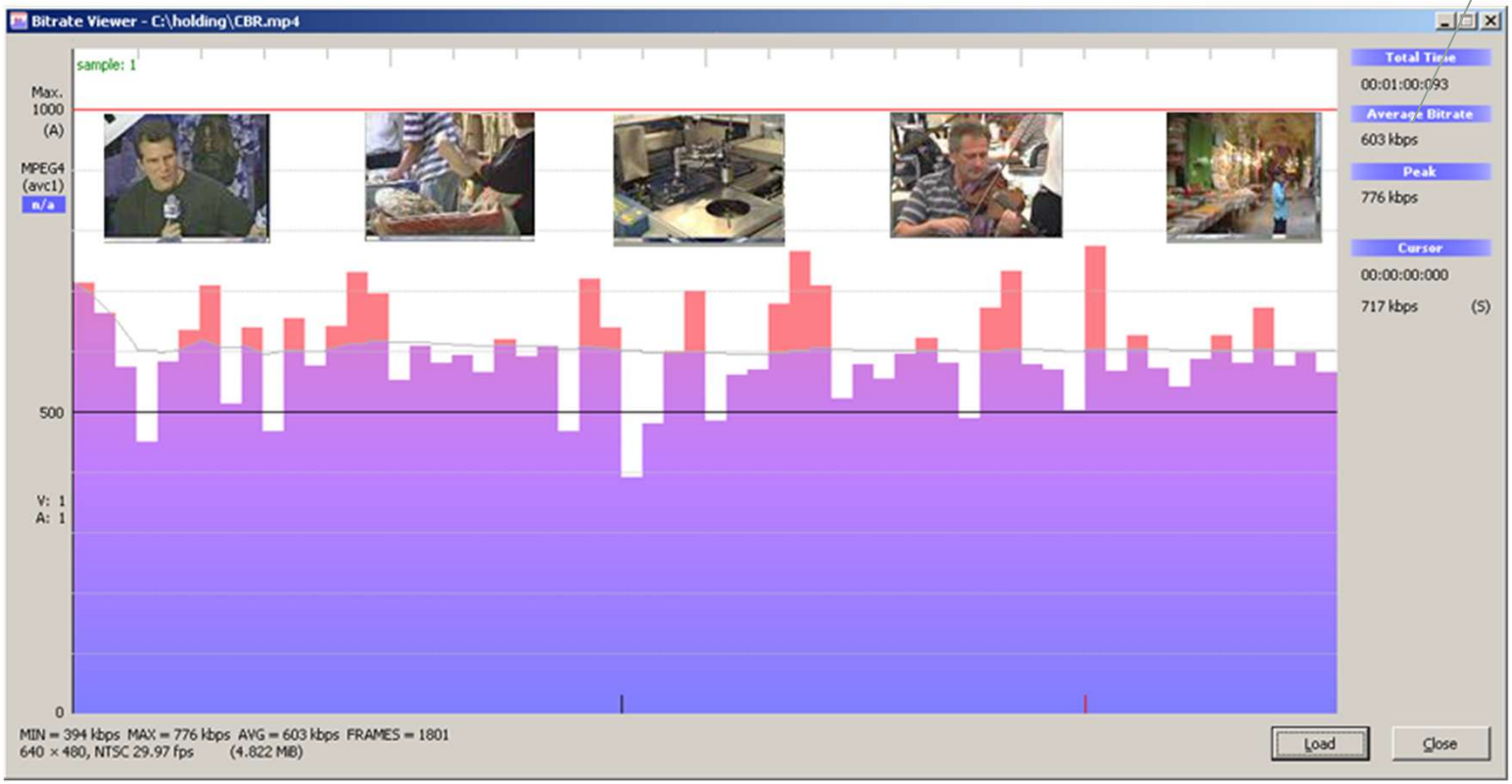
Bitrate Control

- Constant Bitrate (CBR) vs. Variable Bitrate (VBR)
- Producing top quality VBR and CBR
- When to use CBR and VBR

Bitrate Control Alternatives Constant (CBR) vs. Variable Bit Rate (VBR)



CBR File Illustrated



603 kbps Average

- Faint (sorry) wavy blue line is data rate
- Relatively consistent throughout

VBR File Illustrated

596 kbps Average

- Faint (sorry) wavy blue line is data rate
- Varies with scene complexity

Constant Bitrate

- Defined: One bit rate applied to entire video, irrespective of content
- Pros:
  - Computationally easy
  - Fast - one pass will do it
- Cons: Doesn't optimize quality

Variable Bitrate

- Defined: Dynamic bit rate matches motion in video
- Pros: Best quality
- Cons:
  - Need two or more passes
  - Can produce stream with variability

CBR vs. VBR

CBR

- Which file is easier to deliver over fixed bandwidth connections?
  - CBR

VBR

- Which file streams more reliably over changing conditions?
  - CBR

Adaptive - VBR vs. CBR

- Adaptive–most pundits recommend CBR
  - More consistent stream
  - Fewer encoding-related stream switches
- In practice–many producers use ***constrained VBR***
  - Some as high as 200% (MTV)
  - Obviously, they wouldn't if this caused problems

Choosing Between VBR and CBR

- Getting objective
- Overall quality
- Transient quality
- Deliverability

Introduction to Moscow State University VQMT

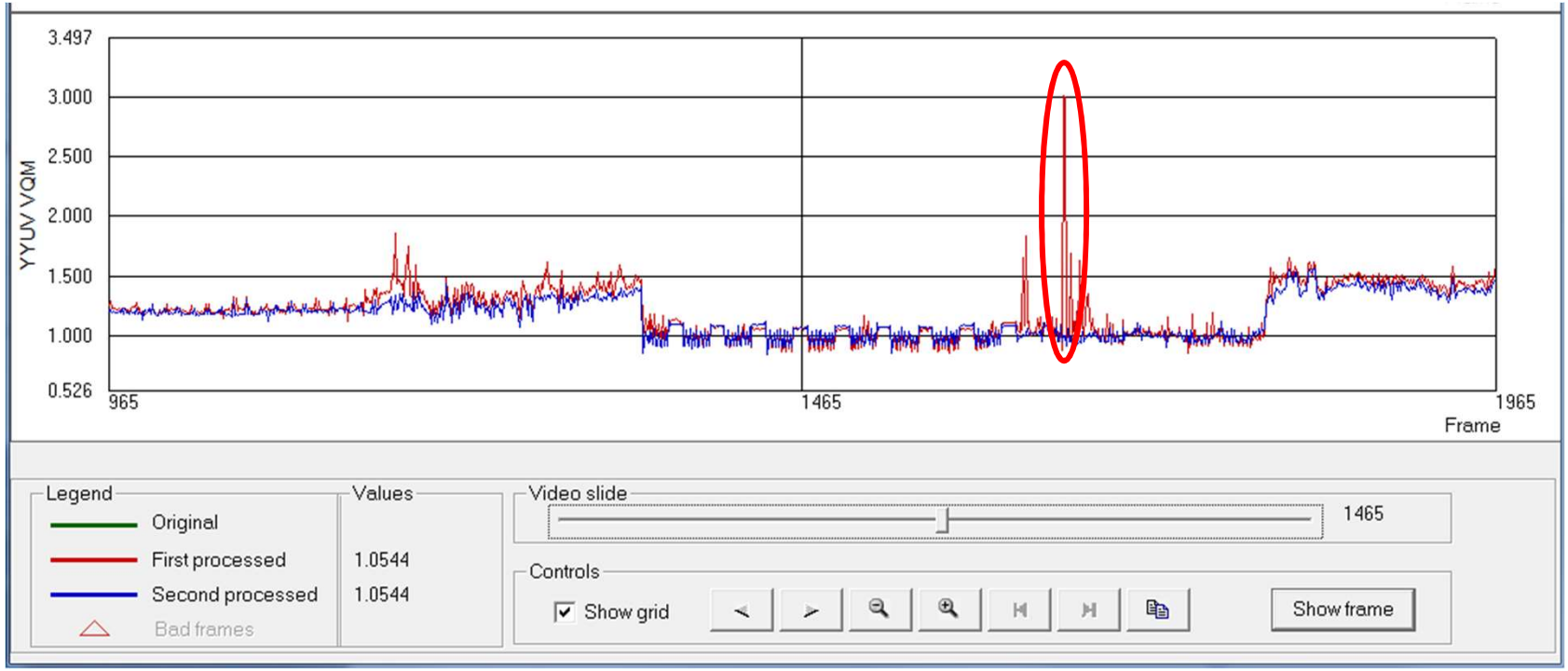
- Video Quality Measurement Tool
  - Input compressed file and original
  - Tool outputs various quality measures
    - **Peak Signal to Noise (PSNR) – Shown primarily in this presentation (higher scores better)**
    - Structured Similarity (SSIM)
    - Video Quality Metric (VQM)
- Can objectify previously subjective analysis
- Can compare alternatives numerically and visually

How Much Better Quality is VBR over CBR?

PSNR	200% VBR	150% VBR	110% VBR	CBR 2-Pass	CBR 1-Pass	Total Quality Delta	Delta - 110% to 200%
Tears of Steel	41.97	41.89	41.60	41.40	41.41	1.36%	0.88%
Sintel	41.34	41.13	40.67	40.56	40.17	2.83%	1.64%
Big Buck Bunny	41.73	40.98	40.00	39.70	40.07	4.88%	4.14%
Talking Head	44.23	44.22	44.17	44.12	44.15	0.25%	0.14%
Freedom	42.06	42.02	41.84	41.83	41.65	0.98%	0.53%
Haunted	42.07	42.07	42.01	41.90	42.06	0.40%	0.15%
1080p Average	**42.23**	**42.05**	**41.71**	**41.58**	**41.58**	**2.46%**	**2.20%**
720p Average	**41.35**	**41.18**	**40.82**	**40.77**	**40.76**	**1.71%**	**1.28%**

- Across the spectrum of different types of content
  - 200% CVBR always the highest
  - CBR always the lowest
- Total quality differential is minimal

Transient Quality Issues (ugly frames)



- Moscow University metric visualization
  - Red is CBR; Blue is VBR
  - Circled area shows very significant quality delta

(Metric is VQM, so higher scores are worse)

Here's that Frame

Analysis

- Transient differences like this are:
  - Much more likely in high motion files with significant scene variability
  - Rare
  - Short (1-2 frames)
- That said, VBR
  - Avoids this problem
  - Produces slightly better quality overall

Deliverability

- Research study
- Compared playback efficiency of CBR and 200% constrained VBR files
  - Mixed talking head and ballet footage
  - Worst case experience
- Restricted playback – used tool called Charles Debugging Suite to limit bandwidth during playback

Video Streams

Codec	Width	Height	Bitrate (kbps)
H.264	1920	1080	4500 (VBR)
H.264	1280	720	3100 (VBR)
H.264	1280	720	2100 (VBR)
H.264	960	540	1500 (VBR)
H.264	640	360	1000 (VBR)
H.264	480	268	550 (VBR)
H.264	320	180	260 (VBR)

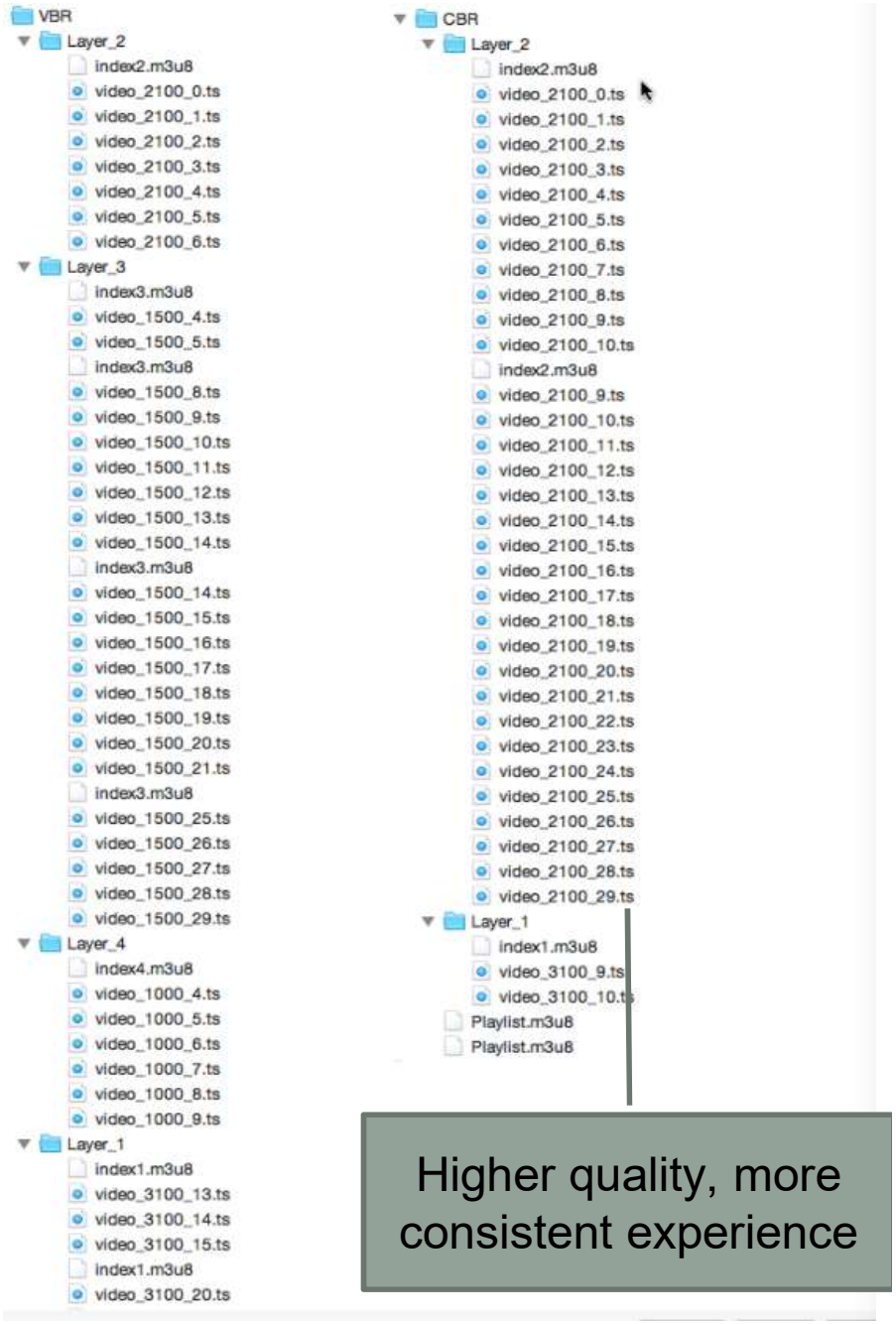
Our Findings

- Throttle to 3200

		VBR - Safari			CBR - Safari		
Layer	**Bitrate**	**Segs**	**%**	**Bandwidth**	**Segs**	**Percent**	**Bandwidth**
0	4500		0%	0		0%	0
1	3100	11	22%	34,100	2	6%	6,200
2	2100	7	14%	14,700	32	94%	67,200
3	1500	22	45%	33,000		0%	0
4	1000	9	18%	9,000		0%	0
5	550		0%	0		0%	0
6	260		0%	0		0%	0
Total		**49**	**100%**	**90,800**	**34**	**100%**	**73,400**

More bandwidth (repeat packets) ($$$)

Better QoE



Higher quality, more consistent experience

Throttle to 4500 – Playback in Safari

- Throttle to 4500

		VBR - Safari			CBR - Safari		
Layer	**Bitrate**	**Segs**	**%**	**Bandwidth**	**Segs**	**Percent**	**Bandwidth**
0	4500		0%	0		0%	0
1	3100	7	14%	21,700	30	88%	93,000
2	2100	25	51%	52,500		0%	0
3	1500	6	12%	9,000		0%	0
4	1000		0%	0		0%	0
5	550		0%	0		0%	0
6	260		0%	0		0%	0
Total		**38**	**78%**	**83,200**	**30**	**88%**	**93,000**

More switching, many lower-quality segments (reduce QoE)

More consistent quality Better QoE

Playback m3u8 in Safari

Throttle to 4500 – JW Player

		VBR - JWPlayer			CBR - JWPlayer		
Layer	**Bitrate**	**Segs**	**Percent**	**Bandwidth**	**Segs**	**Percent**	**Bandwidth**
0	4500		0%	0		0%	0
1	3100	7	18%	21,700	30	88%	93,000
2	2100	25	63%	52,500		0%	0
3	1500	5	13%	7,500		0%	0
4	1000		0%	0		0%	0
5	550		0%	0		0%	0
6	260		0%	0		0%	0
Total		**37**	**93%**	**81,700**	**30**	**88%**	**93,000**

More switching, many lower-quality segments (reduce QoE)

More consistent quality
Better QoE

WHAT APPLE DOES

- Apple says 200% Constrained VBR is OK
  (http://bit.ly/hls_spec_2017)

1.30. For VOD content the peak bit rate SHOULD be no more than 200% of the average bit rate.

- Apple's bitrate ladder ~ 110% CBR

HEVC	Width	Height	FPS	(Peak) Bandwidth	Average Bandwidth	VBR Constraint
1080p_h	1920	1080	60	6,472,228	5,913,163	109%
1080p_m	1920	1080	60	5,235,899	4,609,073	114%
1080p_l	1920	1080	60	3,887,770	3,249,312	120%
720p	1280	720	60	2,572,701	2,443,933	105%
540p	960	540	60	1,972,328	1,774,314	111%
432p	768	432	30	1,034,255	946,612	109%
360p	640	360	30	709,770	637,339	111%
270p	480	270	30	356,927	330,229	108%
234p	416	234	30	148,713	122,941	121%
Average						**112%**

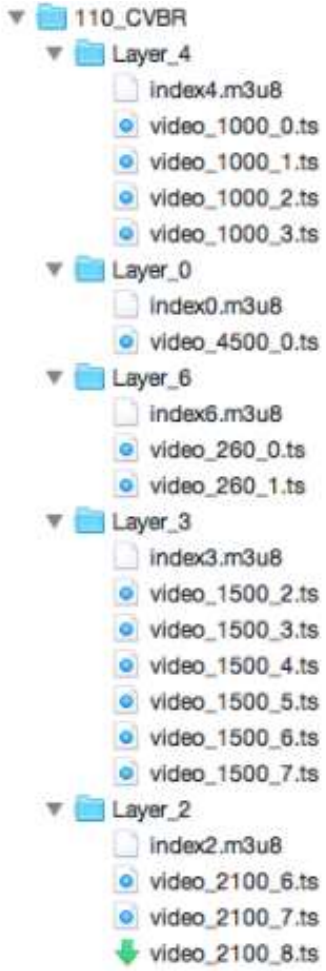
(https://developer.apple.com/streaming/examples/)

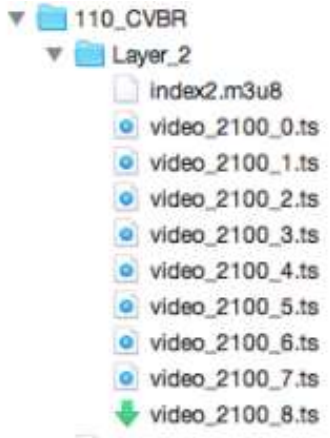
Conclusions

- 200% constrained VBR may reduce QoE when delivered over constrained conditions
- CBR best for overall QoE
  - But, has transient quality issues
  - 110% constrained VBR is the best compromise

Which Layer Should You Play First?

- Description – Same encoded files
  - Left – Layer 0 is first (4500)
  - Right – Layer 2 (2100)
  - Constrain at 3200
  - Play till segment 8
- Observation
  - More packets, lower QoE when choose wrong starting point
- Conclusions
  - First file selected should be sustainable
  - Should change depending upon connection



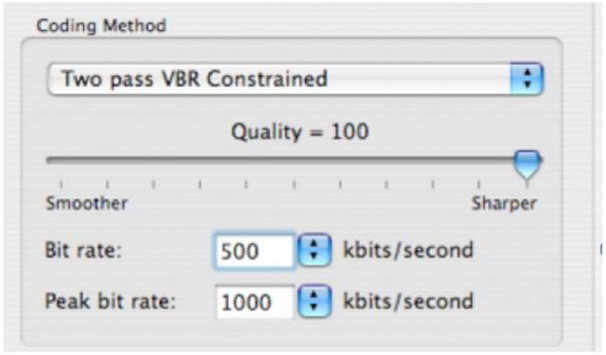


Apple Recommendations for HLS

- Wi-Fi – 2 mbps stream first
- Cellular – 760 kbps stream first

How Does Constrained Work?

- Encoder dependent
  - Set target data rate
  - Set maximum in data rate or % of target
  - Some let you set minimum and maximum
- Rules of thumb
  - Single file streaming
    - Max – 200% of target
    - Min – 50% of target



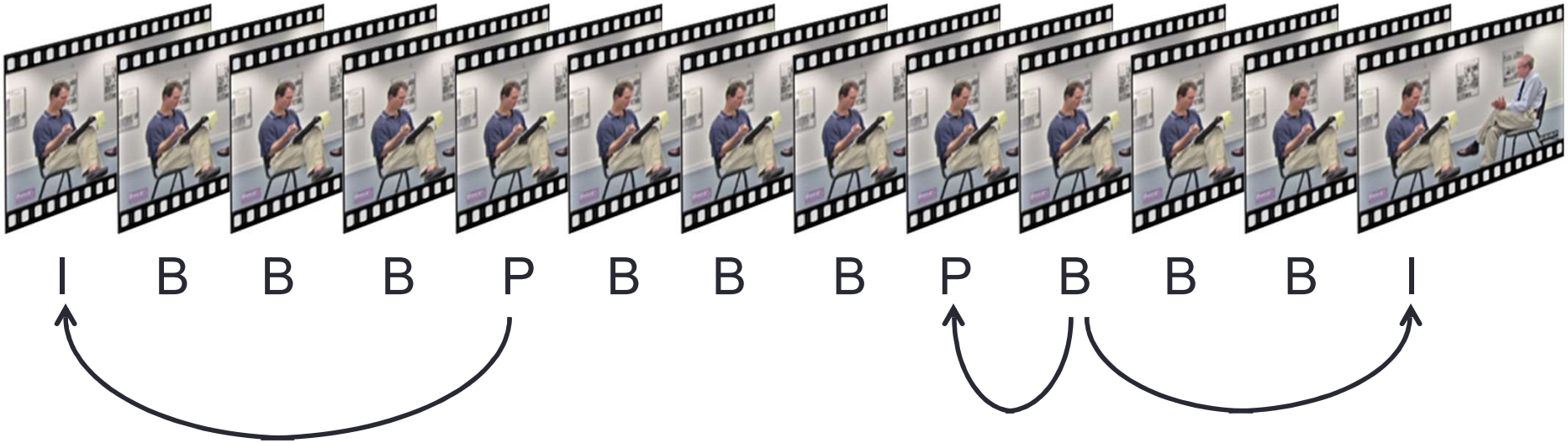
Bottom Line

- CBR is most conservative
  - Of course, use for live
  - lowest quality, transient problems
  - Best deliverability
- 110% constrained VBR is next, preserving switching heuristics and avoiding CBR quality deficits
- Many producers are more aggressive and use 200% constrained VBR

I-, B- and P-frames

- Defining keyframes (used interchangeably with I-frame)
- Choosing a keyframe interval
- Configuring keyframe parameters
- B-frames
- Reference frames

What are I, B and P Frames?



- I-Frame - encoded without reference to other frames (also called keyframes)
- P - looks backward to I and P frames (predictive)
- B - looks forward and backward to previous I and P frames (Bi-directional interpolated)
  - No frames refer to B-Frame (most of the time)

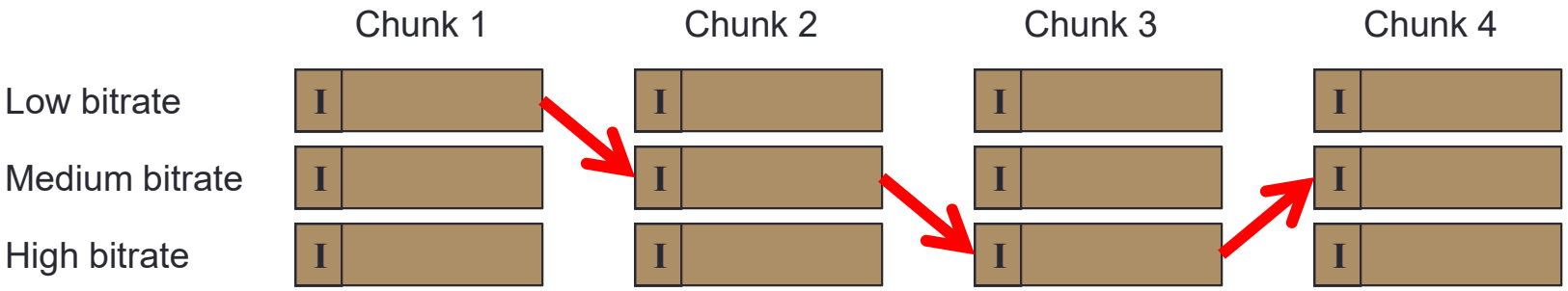
Configuring keyframes

- Though largest frame, keyframes enhance interactivity
  - All playback starts on a keyframe
  - When seeking to random frame (like the third p-frame), must start playback at preceding keyframe
  - To enhance interactivity, maximum key interval should be 5-10 seconds

What About Adaptive?

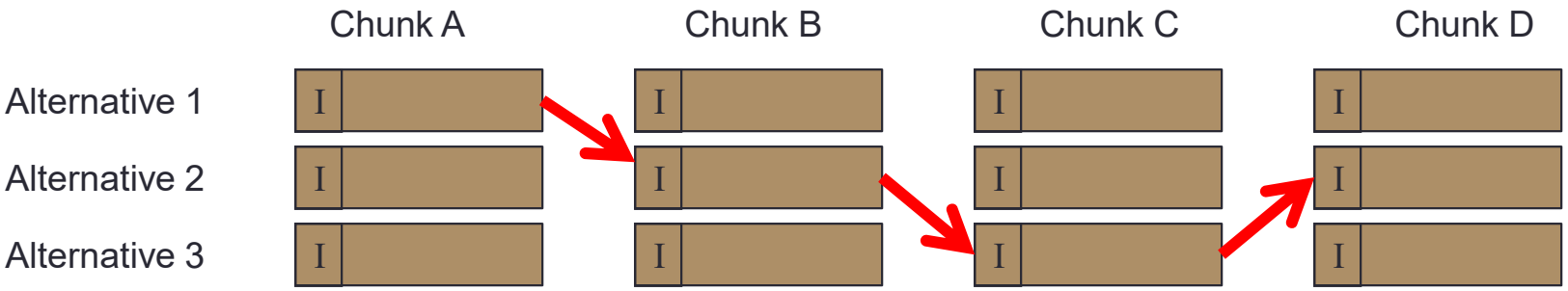
- Rules change when encoding a group of files for adaptive streaming
  - Shorter keyframe interval to enable more nimble stream switching
    - Ten seconds is ***forever*** when bandwidth drops
  - Keyframe interval must match in all files
  - Need regular interval (e.g.. every 90 frames)
    - Disable scene change detection when this will change this interval
  - Needs to divide evenly into segment size

What About Adaptive?



- Adaptive involves multiple streams (low, medium and high) using multiple chunks (1,2,3,4)
  - Switch streams by retrieving chunks from different alternative
  - So, need keyframe (I-frame) ***at start of every chunk***
  - ***So, keyframe interval must equal chunk size or be divisible into chunk size***

What About Adaptive?



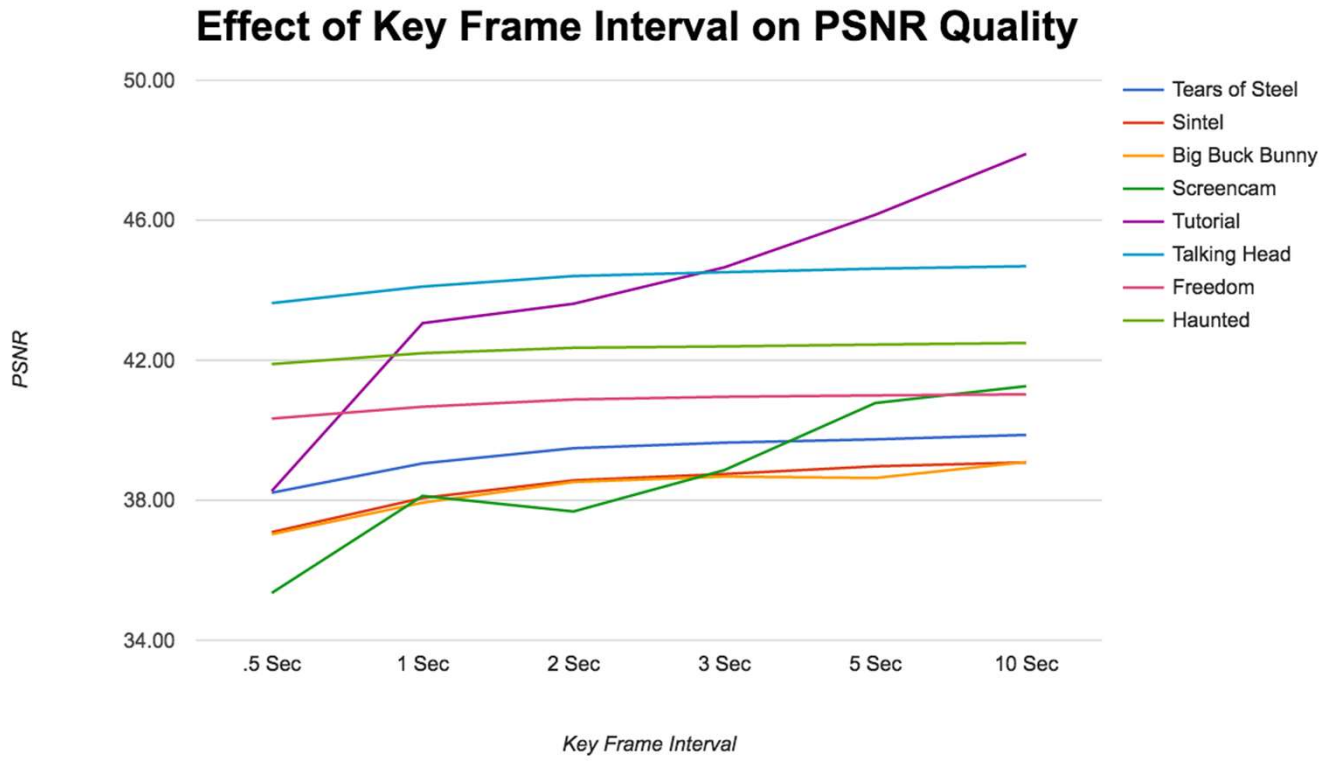
- Need ***regular*** keyframes
  - Some encoders restart keyframe interval when inserting a keyframe at a scene change
  - ***For these, disable scene change detection***, or:
    - Otherwise ensure keyframe at I-frame distance

What's the Best Keyframe Interval

	.5 Sec	1 Sec	2 Sec	3 Sec	5 Sec	10 Sec	Max Delta
Tears of Steel	38.22	39.05	39.49	39.64	39.74	39.87	4.32%
Sintel	37.09	38.06	38.57	38.75	38.97	39.08	5.37%
Big Buck Bunny	37.03	37.93	38.52	38.68	38.64	39.09	5.57%
Talking Head	43.63	44.10	44.40	44.51	44.61	44.68	2.42%
Freedom	40.33	40.67	40.88	40.96	40.99	41.03	1.72%
Haunted	41.89	42.20	42.35	42.39	42.45	42.49	1.44%
Average	**39.26**	**39.96**	**40.37**	**40.51**	**40.59**	**40.75**	**3.88%**
Screencam	35.35	38.13	37.68	38.86	40.78	41.26	16.71%
Tutorial	38.26	43.06	43.61	44.65	46.15	47.89	25.17%

- .5 second never best option
- 2 seconds recommended with good reason

What's the Best Keyframe Interval

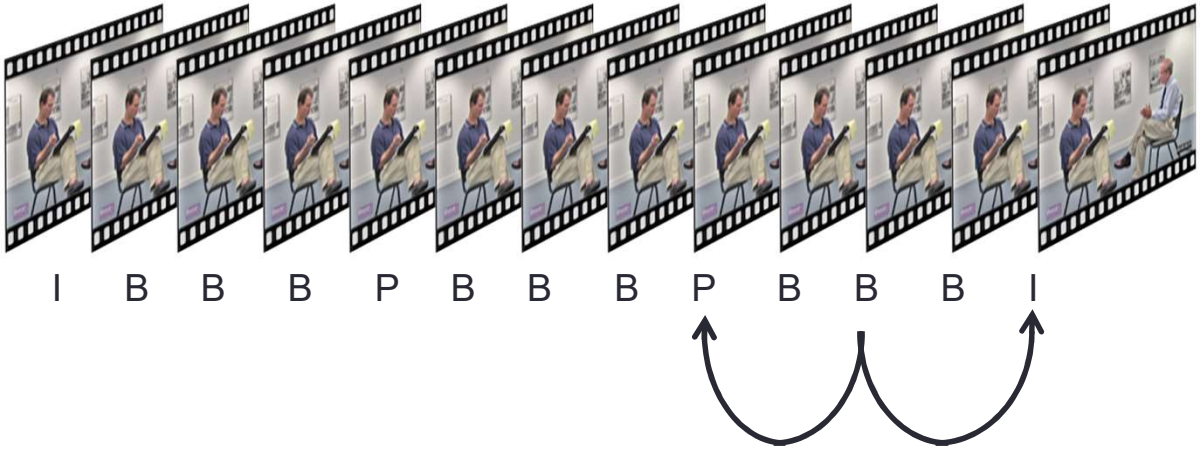


- Not as much difference as you might think
  - Screencam and Tutorial (PPT based video) - outliers
- Real world video, stops increasing after 2-3

Keyframe Summary

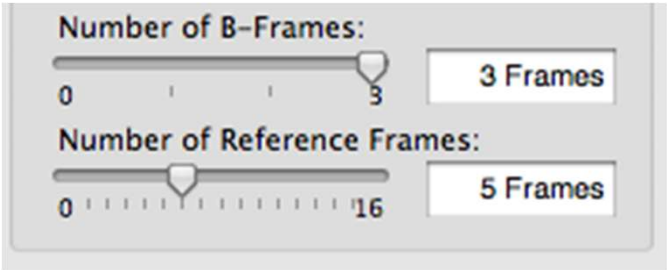
- Single file – 10 – 15 seconds OK
- Adaptive
  - 2-3 seconds
  - Divide evenly into segment size
  - Ensure key frames at start of each segment
    - Disable keyframes at scene changes, or
    - Force keyframes at selected interval

Configuring B-frames



- Main/High profiles only
- Theoretically the most "efficient" frame
  - Should improves quality (comparisons to come)
- Hardest to decode
  - Decoder has to have all referenced frames in memory to decode B-frame
  - Frame usually delivered out of order, which also complicates playback

Typical B-Frame Encoding Parameters



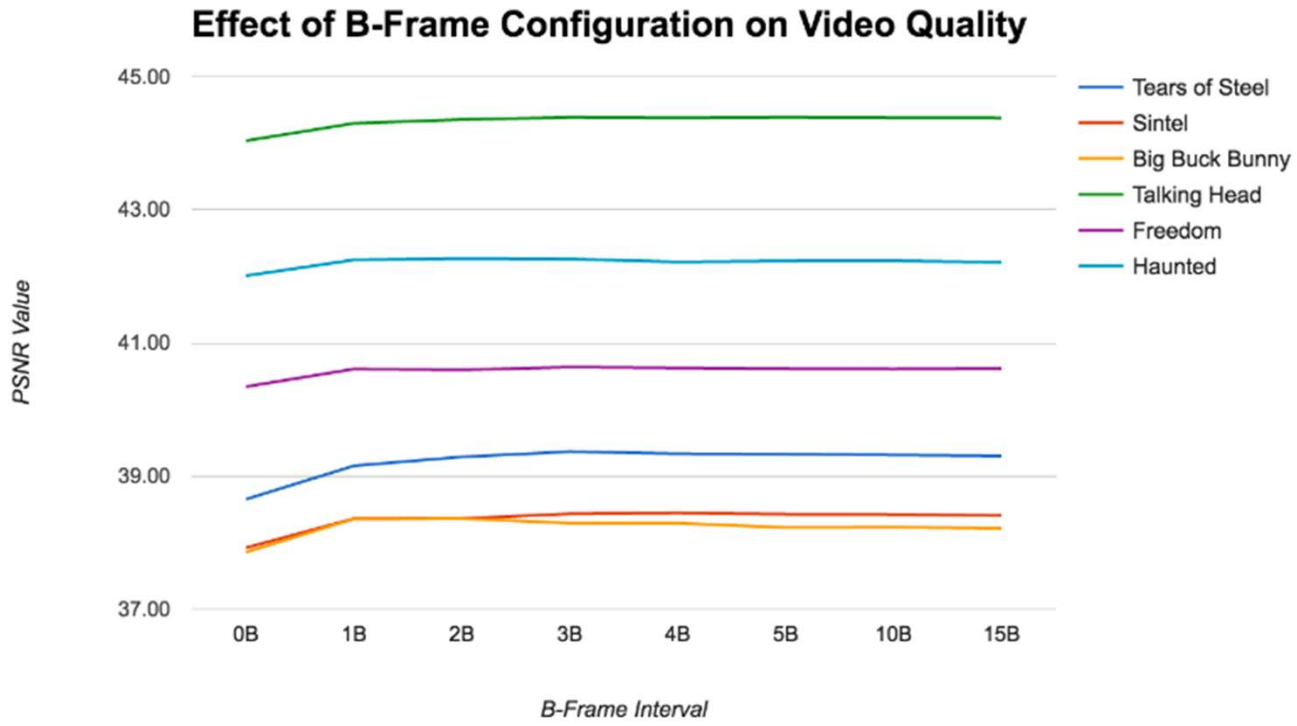
- Number is number of B frames between I and P-Frames; (IBBBPBBBPBBBPBBBP)
  - What's the best value?
- Reference frames (both P and B-frames)
  - Number of frames searched for redundancies
  - What's the best value?

B-frame Tests

	0B	1B	2B	3B	4B	5B	10B	15B	Max Delta
Tears of Steel	38.95	39.41	39.56	39.64	39.64	39.63	39.61	39.64	1.75%
Sintel	38.33	38.71	38.73	38.77	38.75	38.76	38.73	38.72	1.13%
Big Buck Bunny	38.30	38.73	38.79	38.68	38.65	38.63	38.57	38.54	1.27%
Talking Head	44.21	44.43	44.51	44.51	44.51	44.51	44.50	44.50	0.68%
Freedom	40.76	40.93	40.92	40.96	40.93	40.93	40.92	40.91	0.49%
Haunted	42.20	42.33	42.38	42.40	42.37	42.38	42.35	42.36	0.47%
Average	**40.46**	**40.76**	**40.82**	**40.83**	**40.81**	**40.81**	**40.78**	**40.78**	**0.96%**
Screencam	36.94	38.04	38.13	38.66	39.93	39.64	39.00	38.67	7.49%
Tutorial	45.02	45.09	44.91	44.11	44.46	44.16	44.62	44.41	2.18%

- For most files, quality peaks at 3
- Also not a significant difference either way (probably not noticeable

B-frame Tests



- For most files, quality peaks at 3
- Minimal improvement after 1

Zencoder (Cloud Encoder)

- “The default is 0 for widest compatibility. We recommend a value of 3 for compression/quality improvements. Values higher than 5 or 6 rarely provide much benefit, and greatly increase encoding time.”
- (bit.ly/Zen_B).

Quick Summary: B-Frames

1. Always enable adaptive B-frames when available.
2. Always enable B-frames as reference frames when available.
3. Always enable pyramid B-frames when available.
4. A B-frame interval of 3 presents a good mix of optimum quality

Reference Frames

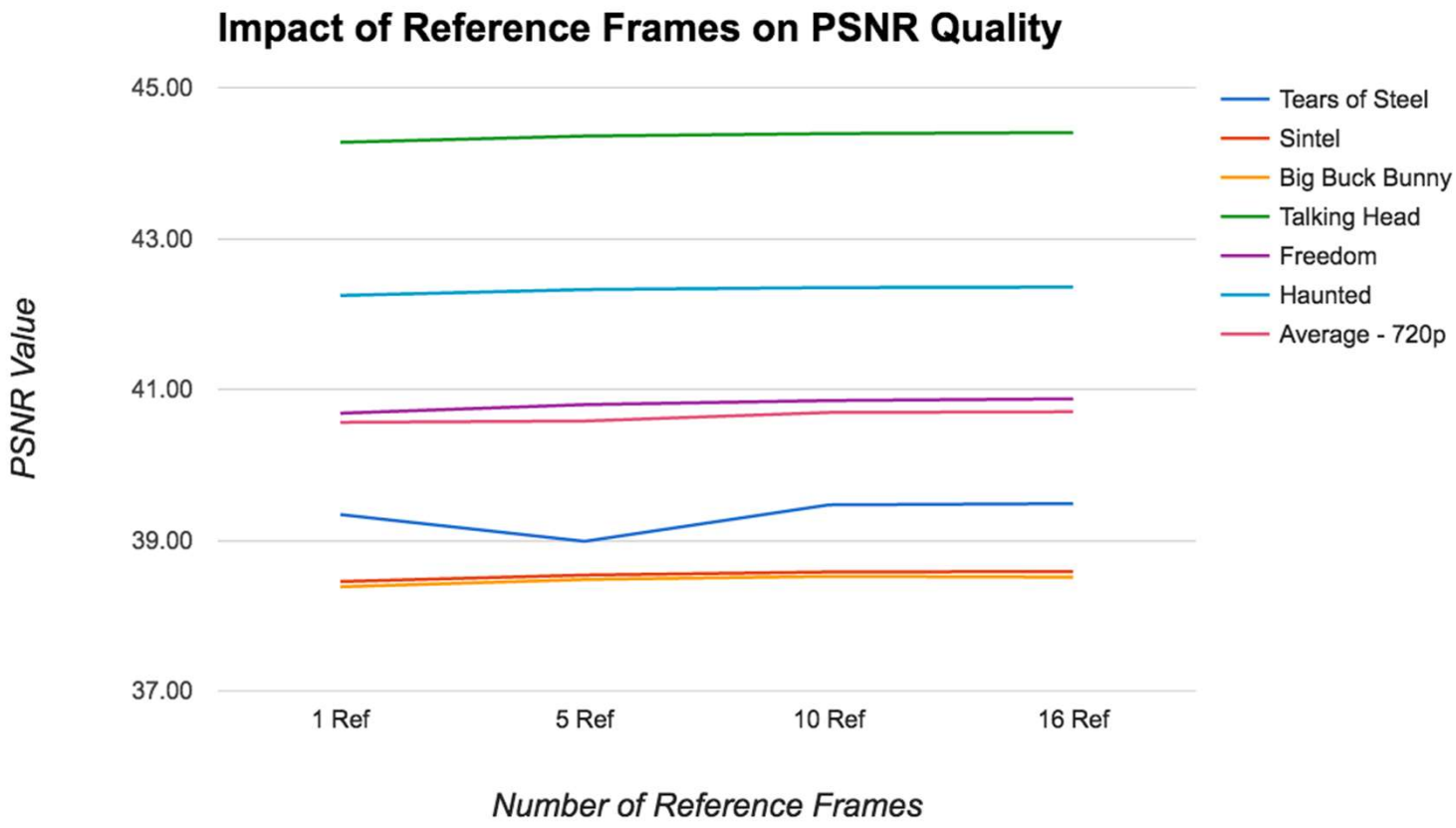
- What are they?
  - Frames from which the encoded frame can find redundant information
- What's the trade-off?
  - Searching through more frames takes more time, lengthening the encoding cycle
  - Since most redundancies are found in frames proximate to the encoded frame, additional reference frames deliver diminishing returns

How Much Quality?

720p-110CVBR	1 Ref	5 Ref	10 Ref	16 Ref	Max Delta	10 - 16 Delta	16 - 5 Delta
Tears of Steel	39.34	38.99	39.47	39.49	1.28%	-0.04%	-1.26%
Sintel	38.45	38.54	38.58	38.59	0.35%	-0.02%	-0.12%
Big Buck Bunny	38.38	38.48	38.52	38.51	0.36%	0.03%	-0.08%
Talking Head	44.27	44.36	44.39	44.40	0.29%	-0.03%	-0.10%
Freedom	40.68	40.80	40.85	40.87	0.47%	-0.06%	-0.19%
Haunted	42.24	42.32	42.35	42.36	0.26%	-0.02%	-0.08%
Average - 720p	**40.56**	**40.58**	**40.69**	**40.70**	**0.34%**	**-0.02%**	**-0.30%**

- 16 is best
  - Miniscule difference between 16 and 10 (.02%)
  - .3% delta between 5 and 16

How Much Quality?



- Very little notable difference

How Much Time?

Encoding Time	1 Ref	5 Ref	10 Ref	16 Ref	Max Delta	10 - 16 Delta	16 - 5 Delta
Tears of Steel	39	49	72	91	133%	-21%	-46%
Sintel	40	53	71	76	90%	-7%	-30%
Big Buck Bunny	41	53	68	85	107%	-20%	-38%
Talking Head	37	47	61	77	108%	-21%	-39%
Freedom	99	142	200	263	166%	-24%	-46%
Haunted	47	65	93	123	162%	-24%	-47%
Average - 720p	**51**	**68**	**94**	**119**	**136%**	**-21%**	**-43%**

- 16 is ~ 2.5 x longer than 1 reference frame
  - Cutting to 5 reduces encoding time by 43% (close to doubling capacity
  - Reduces quality by .3%

Reference Frames

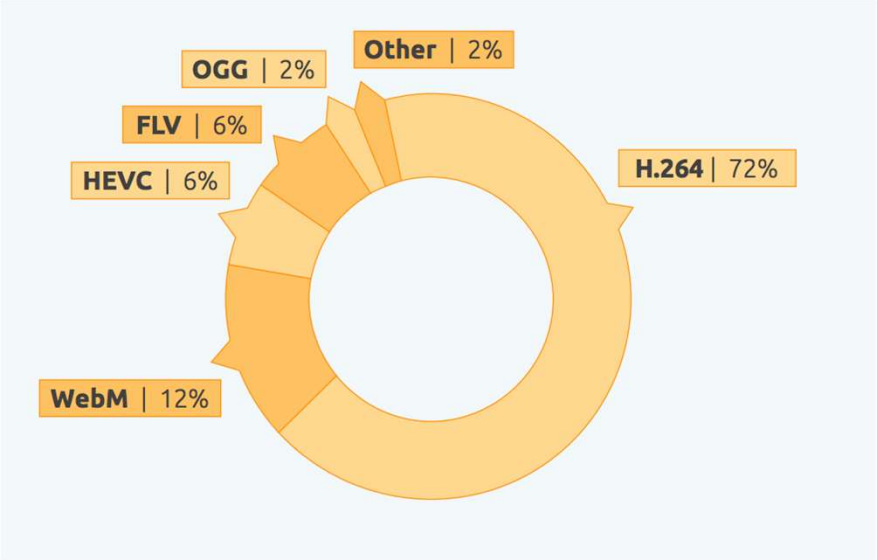
- Recommend 5 as best blend of performance and quality
  - Dropping lower improves performance time only slightly
  - Going higher increases quality negligibly but can dramatically impact encoding time

Choosing a Codec

- Codec – stands for enCOde/DEcode
  - Need the decode side to play the video
- Which platforms have decoders?

	Computer/ Notebook	iOS	Android	Retail OTT (Roku, Apple TV)	Smart TV
H.264	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
HEVC	**MacOS/Windows 10 with h/w**	**Yes**	**Version 5+**	**Most**	**All 4K**
VP9	**Chrome, Firefox, Opera, Edge**	**No**	**Version 4**	**Some**	**Many Newer**

2015 Numbers from encoding.com

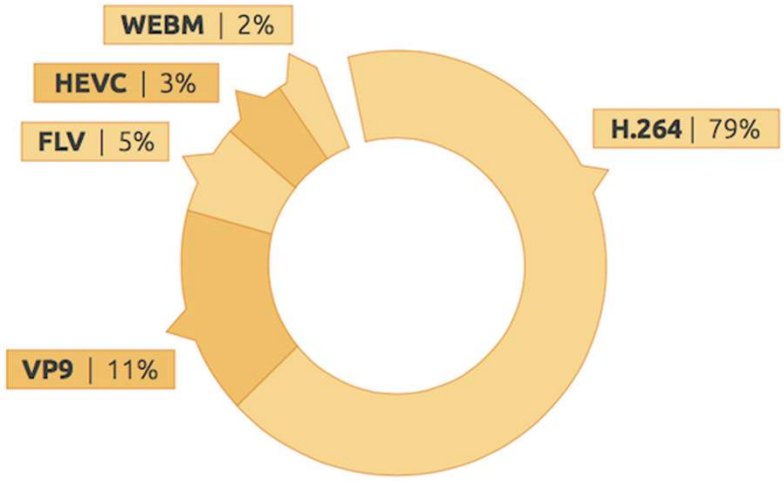


encoding.com 2016 Global Media Format Report
http://bit.ly/ecmf2016

- H.264 still king
- HEVC coming
- WebM includes both VP8 and VP9

- FLV – Flash specific is going away (but Flash can play MP4 files)

2016 Numbers from encoding.com



encoding.com 2017 Global Media Format Report
http://bit.ly/ecmf_2017

- H.264 still king (dropped 7 % points)
- HEVC dropped from 6% - 3%
- VP9 debuts at 11%
- FLV – Flash specific is going away (but Flash can play MP4 files)

The Shape of Things to Come

USB media formats

Video — H.264/AVC (.MKV, .MP4, .MOV), on Roku 4 only: H.265/HEVC (.MKV, .MP4, .MOV); VP9 (.MKV)

Video Codec
H.264 BP/MP/HP
HEVC (H.265 - Main, Main10, Main4:2:2 10)
Motion JPEG
MVC
DivX 3.11 / 4 / 5 / 6
MPEG4 SP/ASP
Window Media Video v9(VC1)
MPEG2
MPEG1
Microsoft MPEG-4 v1 , v2 , v3
Window Media Video v7(WMV1),v8(WMV2)
H 263 Sorenson
VP6
VP8
VP9
RV8/9/10 (RV30/40)

Many new devices will support both HEVC and VP9

Codecs: What's Also Here

- V-Nova Perseus – UK Company
  - Targets
    - Upgrade H.264-based STBs (Sky Italia)
    - Very low bandwidth video (FastFilmz in India)
    - High end contribution
  - Current status re: general purpose streaming
    - Not yet available within the browser for computers or mobile, though they are working on this

Codecs: What's Coming?

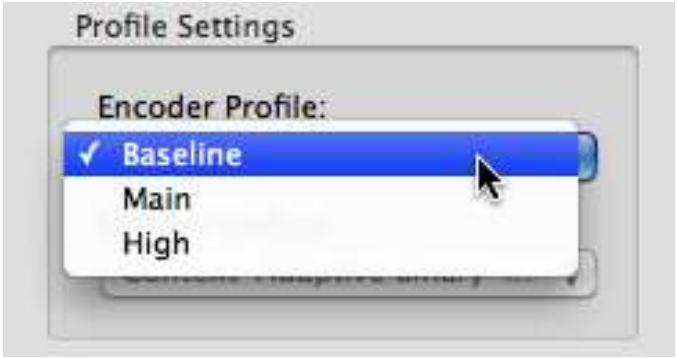
- AV1 Codec
  - Alliance for Open Media (Google, MS, Mozilla, Cisco, Intel, Amazon, ARM, AMD, NVIDIA, Netflix, Amazon)
  - Consolidate open source development from Google (VP10), Mozilla (Daala) and Cisco (Thor)
    - Bitmovin demonstrated live AV1 at NAB; here at show
    - AV1 bitstream should freeze around 12/2017
      - Expect: Immediate support in Chrome, Edge, Firefox
      - Expect: Immediate use by Netflix/YouTube
      - Expect: Hardware support in 12-18 months (normal cycle)
    - Clear open-source successor to VP9 for HTML5

Use H.264 for Compatibility

- Primary focus is H.264
  - Most broadly compatible
- HEVC – 4K TV sets, particularly for HDR
- VP9
  - Starting to get some traction
    - YouTube (of course)
    - JW Player – all purpose
    - Netflix – low bitrate downloads for Android, moving to browser based deployment

Using H.264: Watch Profiles and Levels

- Profiles/Levels
  - Most critical ***compatibility-related*** setting
    - Encode using wrong profile, file won't play on target device
  - Available on all encoding tools
- Don't exceed profile and level of target device
  - Exclusively a concern with mobile (we'll examine later)
  - Computers and OTT devices can play High profile (any level)

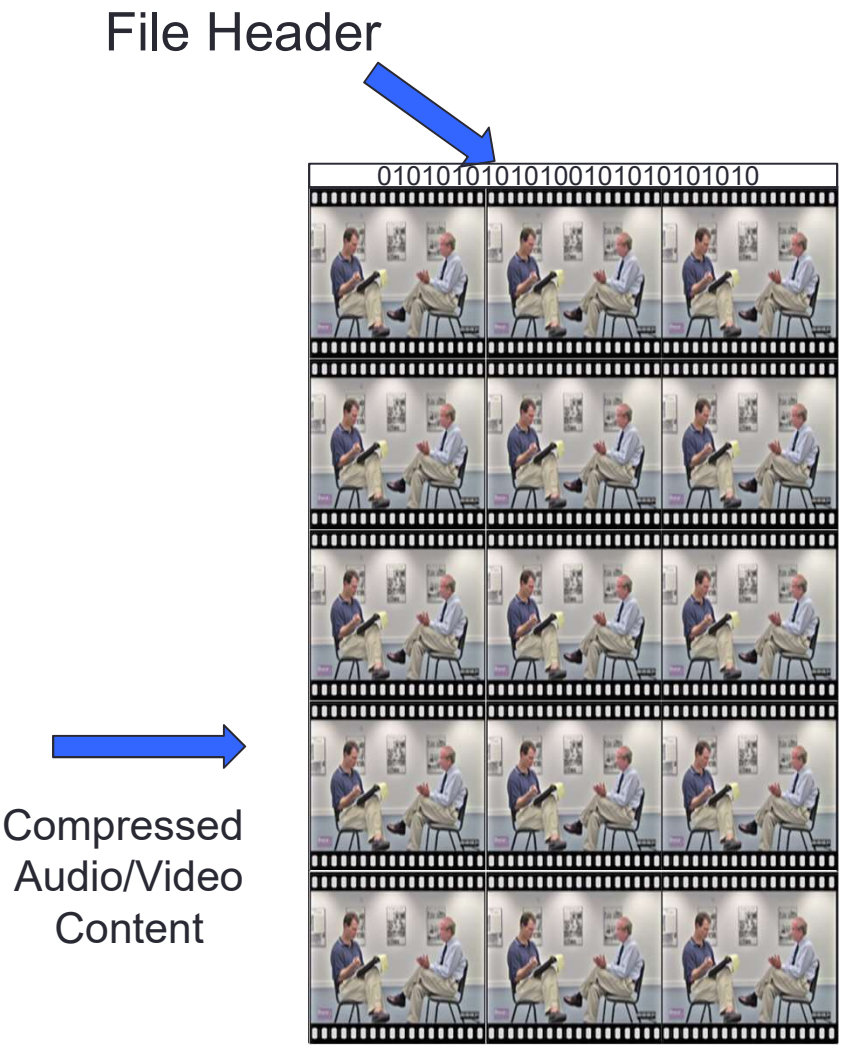


Codecs and Container Formats

- **Codecs:** Compression technologies
  - H.264, VP9, HEVC
- **Container formats**
  - Specs detailing how data/metadata are stored in a file
    - MP4, WEBM, .MPD, .TS, .ISMV, .F4F
  - Also called “wrappers”
    - As in, “encoded the file using the H.264 codec in a QuickTime wrapper”
- Why important?
  - File must be in proper container format to play on target platforms

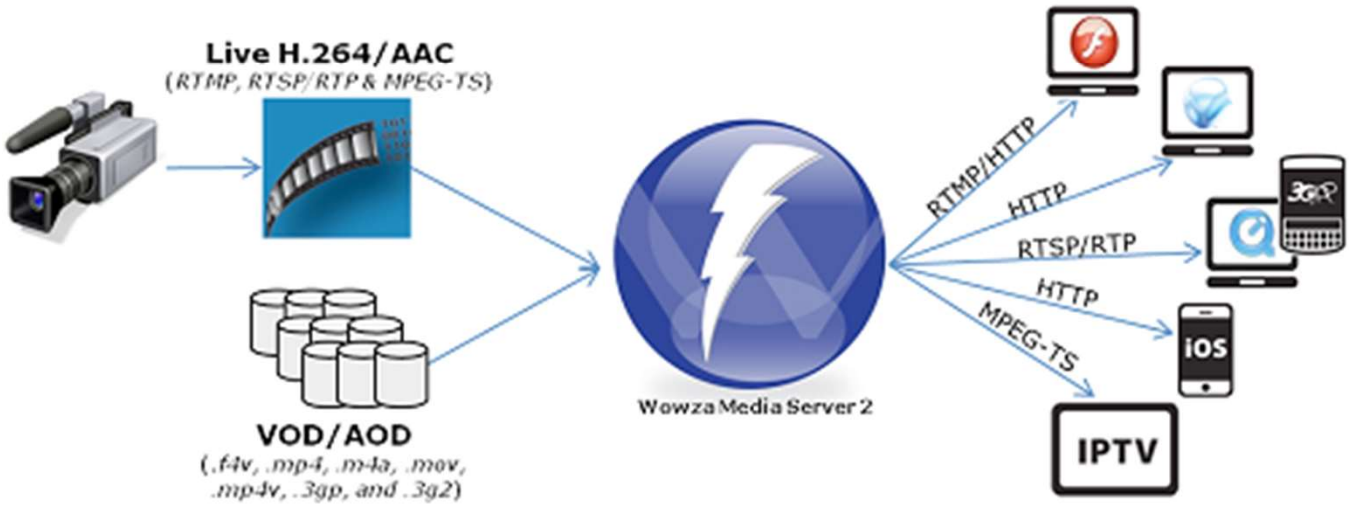
Where is Container Format?

- It's in the file header
  - Very small percentage of overall content
- Can quickly change the container format without affecting A/V content
  - Called ***transmuxing***
  - ***Real Time Packaging***
  - Critical to operation of tools like Wowza Streaming Engine



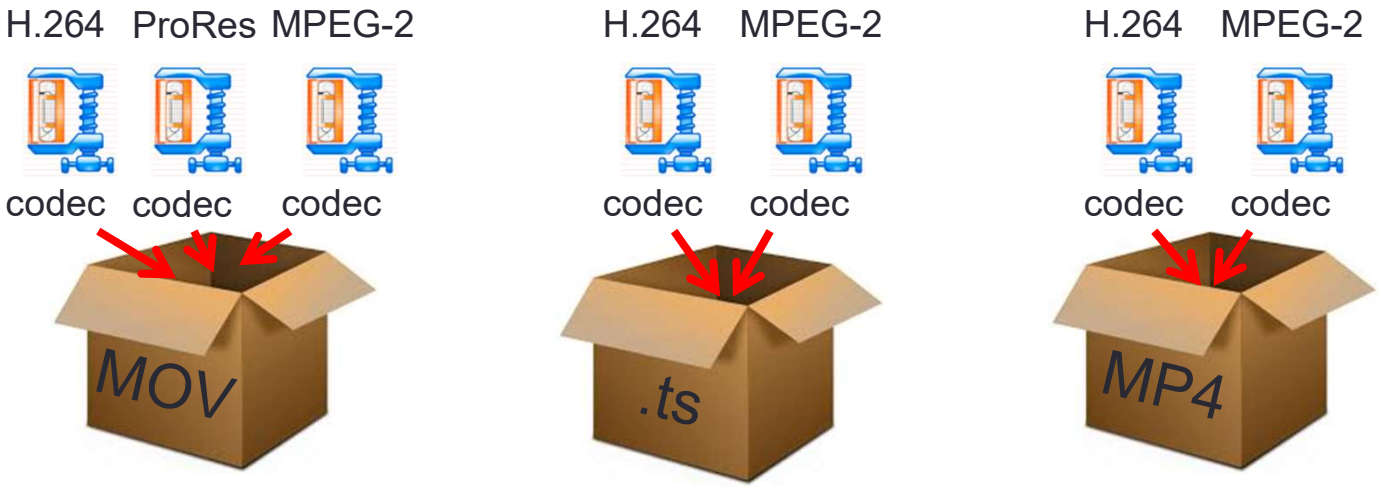
What's Transmuxing Look Like?

- Single format stream in
- Multiple format output streams
  - Why so fast and efficient?
    - Just adjusting file header
    - Not changing compressed video data at all



Key Point on Container Formats

- Separate and distinct from choice of codec
  - Can store MPEG-2 compressed video in MP4 file
  - Can store H.264 video in MPEG-2 transport stream



- Whenever you configure encoder for streaming, be aware of selected codec ***and*** container format

Fundamentals

- Questions?
- Should be 10:00 - ish

Your Target Platforms

- Desktops/Notebooks
- Mobile
- Retail OTT
- Smart TV

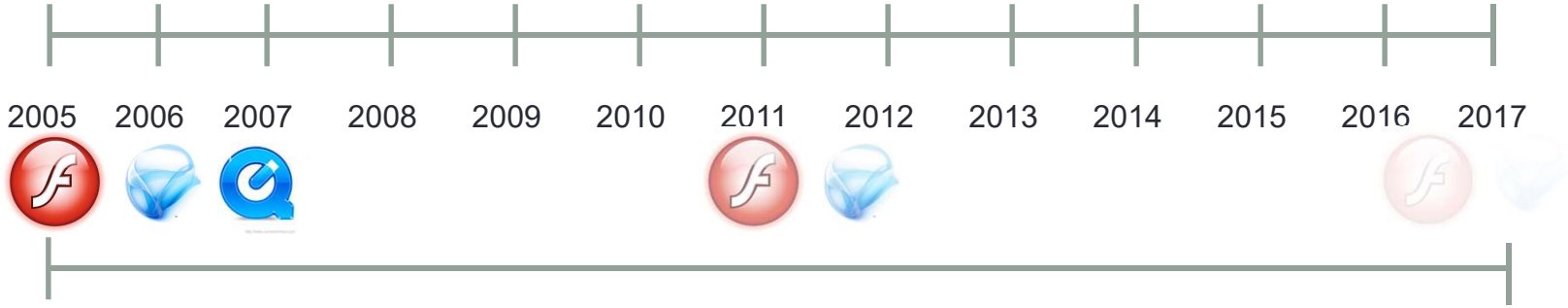
Targeting Desktops and Notebooks

- Working in the HTML5 environment
  - Where we are today
  - What it means
- MSE, EME, and DASH
  - MSE – adaptive, live and captions to HTML5
  - DASH – the MSE compatible delivery format
  - EME – securing MSE streams
- Development alternatives
  - Our goal
  - Our toolsets

Working in the HTML5 Environment

- HTML5's key benefit
  - Video playback without plug-ins
- How it works
  - Instead of obtaining decoders for H.264 and other codecs from plug-ins like Flash/Silverlight
  - Browsers supply players and decoders
    - Decoders can be in the browser (Chrome, Safari, IE)
    - Decoders can be in the OS (Firefox, Opera)
- You're only as good as the deployed browser
  - Can be a problem for services targeting corporate, government or older viewers (check log files)

HTML5 – Where We Are Today



Plug-in era – primarily used now for advertising support

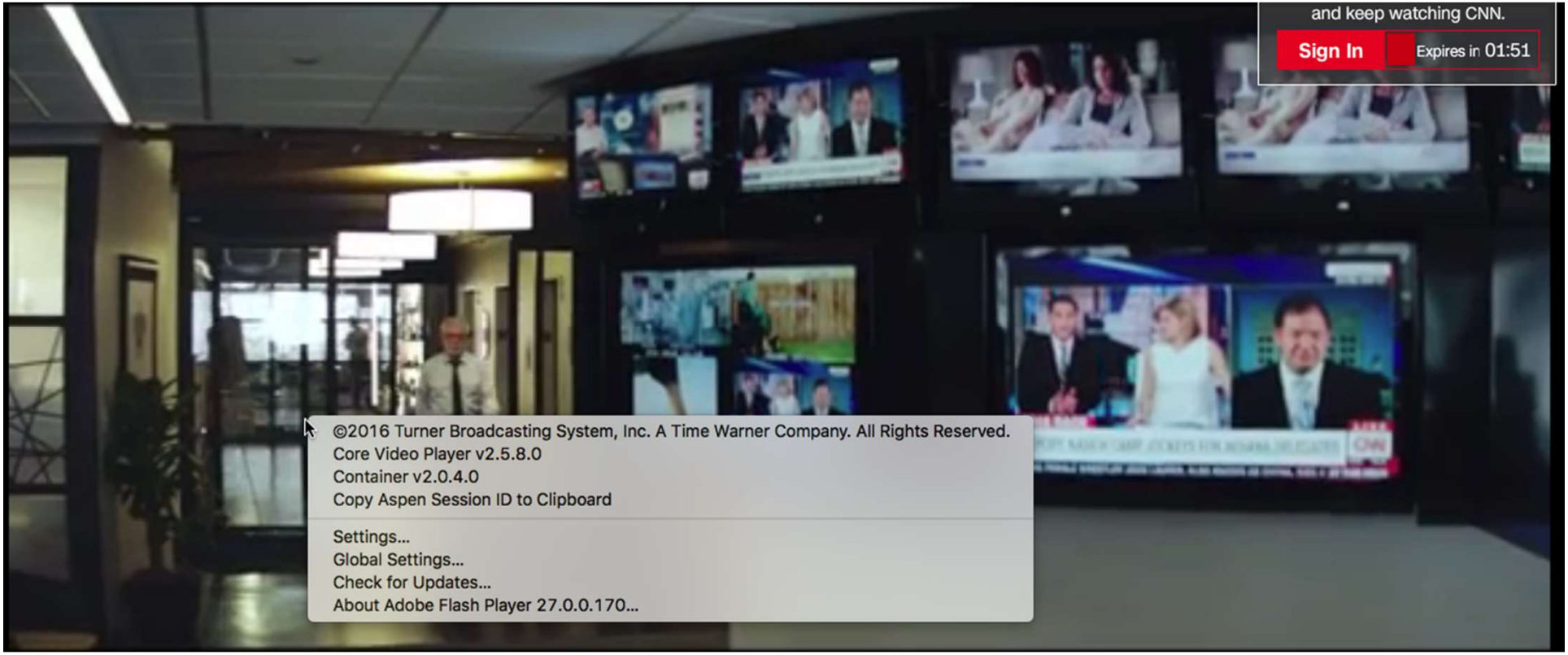
HTML5 Gen 1 <Video> single file, no ABR, no captions, no live
(Flash is Dead, V1)

HTML5 Gen 2 Media Source – Flash equivalent Extensions (Flash is Really Dead)

Some Sites Using HTML5 - ESPN

A look at next year's QBs. Todd McShay's mock draft is out, and two things are clear: It's never too early to look ahead, and the quarterback class is deep.

Some Still With Flash



- CNN Live

HTML5 Generation 1

- Generation 1, circa 2010
  - AKA the video tag <video>
  - The original “Flash killer”
    - No adaptive, live, DRM, captions or advertising support
  - Minimal adaption by premium content distributors
- Lots of smaller websites did transition; *Streaming Media* survey, June 2015
  - Percentage of total streams distributed to desktops via HTML and Flash
    - Those spending $500K annually on encoding/and under $50K

Flash vs. HTML5	2015		
	All	$500K+	$50K-
Flash	53%	52%	52%
HTML5	47%	48%	48%

HTML5 Generation 2 – 2014-2015

- Media Source Extensions (MSE)
  - Live, adaptive streaming, captions
  - Plays files produced in DASH file format
- Encrypted Media Extensions (EME)
  - Browser-based DRM

Understanding HTML5 Gen 2

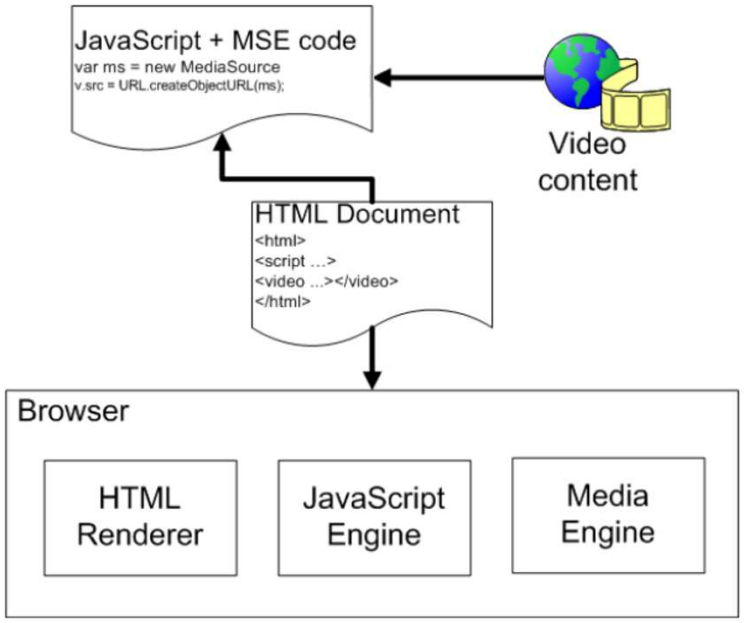
- Pieces of the puzzle
- Implementing MSE
- Implementing EME
- Developmental alternatives

Pieces of the Puzzle

- Media Source Extensions - MSE
- Dynamic Adaptive Streaming over HTTP - DASH
- Encrypted Media Extensions - EME
- Common Encryption - CENC
- ISO-Base Media File Format - BMFF

Media Source Extensions (MSE)

- A W3C HTML Working Group spec
- JavaScript interface to play back media data provided by the JavaScript layer
- More flexible than video tag
  - Media chunks (adaptive) and (closer to) true streaming than progressive
  - Live
  - Better support for captions and DRM (via Encrypted Media Extensions)



What is Dynamic Adaptive Streaming over HTTP (DASH)

- Standardized file format
  - HLS, Smooth, HDS all proprietary
- Like all HTTP-based technologies, it has
  - Fragmented video chunks (or single file with segments)
  - Manifest files

What is DASH? CA$H!

DASH

- IP History
- What do we know?
  - MPEG LA Patent Pool
  - Classes of royalty bearing players
  - Terms
  - Examples

IP History

- MPEG DASH finalized in 2011-2012
- What is MPEG DASH article in Streaming Media (http://bit.ly/what_is_DASH)
  - “Many of the participants who are contributing intellectual property to the effort—including Microsoft, Cisco, and Qualcomm—have indicated that they want a royalty-free solution. While these three companies comprise the significant bulk of the IP contributed to the specification, not all contributors agree, so the royalty issue is unclear at this time."
  - July 2015, MPEG LA announces pool (http://bit.ly/DASH_pool_formed)
  - In November 2016, MPEG LA announces license (http://bit.ly/DASH_license)

MPEG LA License Terms – **DASH Clients**

- DASH Clients (products capable of parsing a Media Presentation Description and accessing or playing DASH Segments)
  - 0 - 100,000 units/year = no royalty (available to one Legal Entity in an affiliated group)
  - US $0.05 per unit after first 100,000 units each year
- What's a DASH client?
  - Multimedia players that play DASH (exoplayer, VLC)
  - Browsers with DASH playback (Edge)
  - Players like JWPlayer, Bitmovin, ***but license currently excludes players that are loaded temporarily through the browser***, though this will be evaluated every 12 months
- Who owes royalty?
  - The company that ***actually supplies*** the player to the end user.

http://bit.ly/DASH_terms

MPEG LA License Terms – DASH Initiators

- DASH Initiators (***essentially apps***)
  - 0 - 100,000 units/year = no royalty (available to one Legal Entity in an affiliated group)
  - US $0.05 per unit after first 100,000 units each year
- What's a DASH initiator?
  - Apps on smartphones, tablets, smart TVs and the like
- Who owes royalty?
  - The company that creates the app
- Royalty cap
  - $30 million combined

http://bit.ly/DASH_terms

Scenarios

- From article MPEG-DASH Royalties: What we know so far (bit.ly/cash4dash)
- 1. Microsoft Edge plays DASH files. That makes it a DASH Client and a per-unit royalty is due.
  - **MPEG LA response:** Assuming Edge plays DASH files, yes.
- 2. All other current browsers support MSE, but can't parse, access, or play DASH files. These are not DASH Clients so no royalty is due.
  - **MPEG LA response:** Probably correct, but we will evaluate on a case-by-case basis.
- 3. If third-party players (JW, Bitmovin, dash.js), are loaded temporarily, they are excluded today, though this may change.
  - **MPEG LA response:** Correct

Scenarios

- 4. An Android phone comes with ExoPlayer, which is a DASH Client. The phone seller pays the royalty.
  - **MPEG LA response:** Correct
- 5. The same Android phone comes with multiple apps that call ExoPlayer to play MPD files. There are DASH Initiators generating a license payable by the app seller.
  - **MPEG LA response**: Correct.
- 6. DASH Players on Smart TVs/OTT devices are DASH Clients. This gives rise to a royalty that the hardware vendor pays.
  - **MPEG LA response:** Correct – player in OS, browser, or a standalone player not associated with an app.
- 7. Apps on Smart TVs/OTT devices. These are DASH Initiators, giving rise to a royalty payable by the app seller.
  - **MPEG LA response:** Correct.

Scenarios

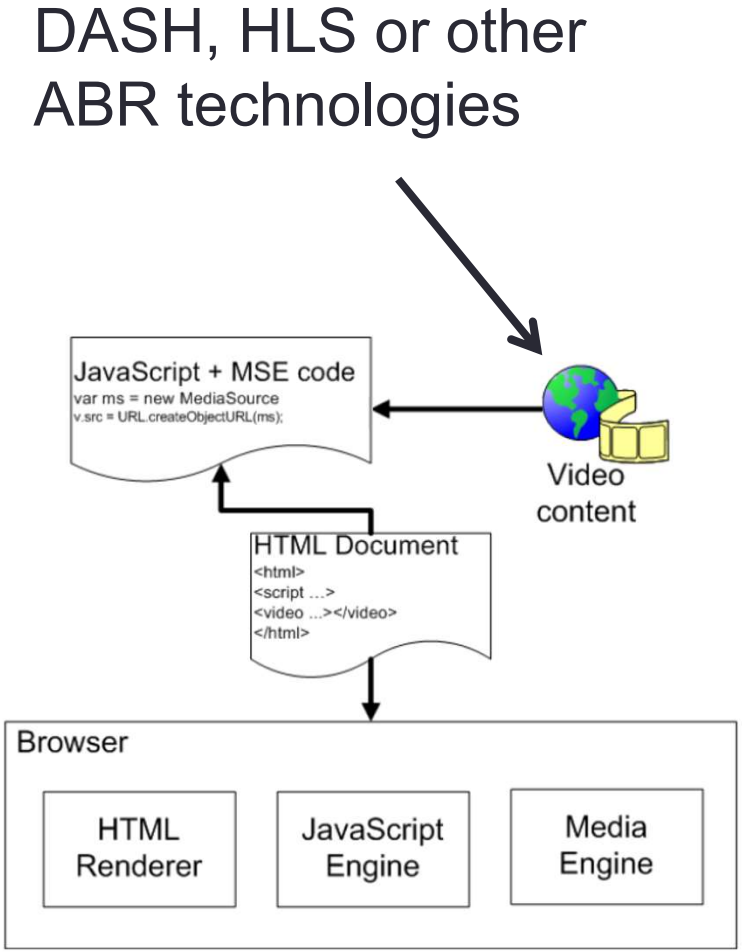
- 8. A consumer downloads an app after purchasing a hardware device. This is a DASH Initiator, giving rise to a royalty payable by the app seller.
  - **MPEG LA response:** Correct.
- 9. I have a Netflix account and apps on six devices. These are all DASH Initiators, and Netflix owes six royalties.
  - **MPEG LA response:** Correct

Analysis and Implications

- This is the first royalty on free internet video
- CNN distributes free video in H264 or HEVC using HLS
  - No royalty
- CNN distributes free video with DASH
  - Royalty on apps and ultimately perhaps browser-based playback
- No exclusions for churches, charities, governments or otherwise
- Really is remarkable in scope

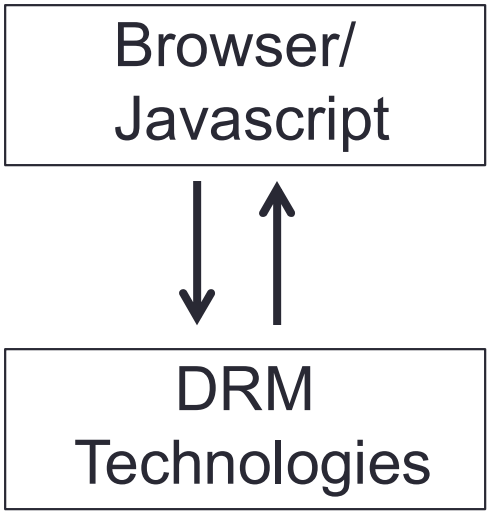
DASH and MSE

- DASH is one of the file formats MSE expects
- Can write JavaScript code enabling MSE to play HLS and other ABR formats
  - Very common among off the shelf players



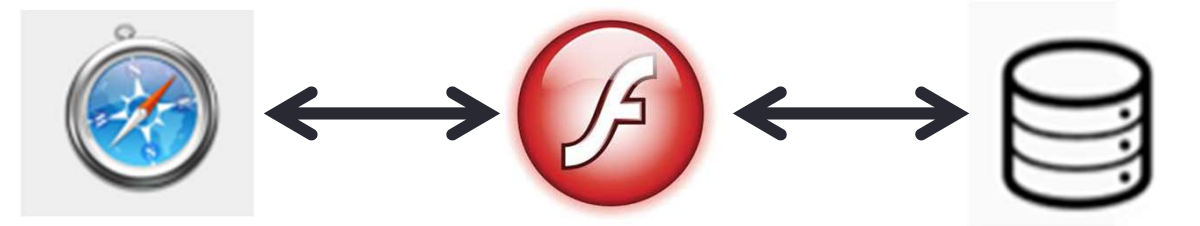
Encrypted Media Extensions (EME)

- JavaScript API
  - Enables HTML5-based digital rights management (DRM)
  - Extends MSE by providing APIs to control playback of protected content.
- License/key exchange is controlled by the browser
  - Not a plug-in



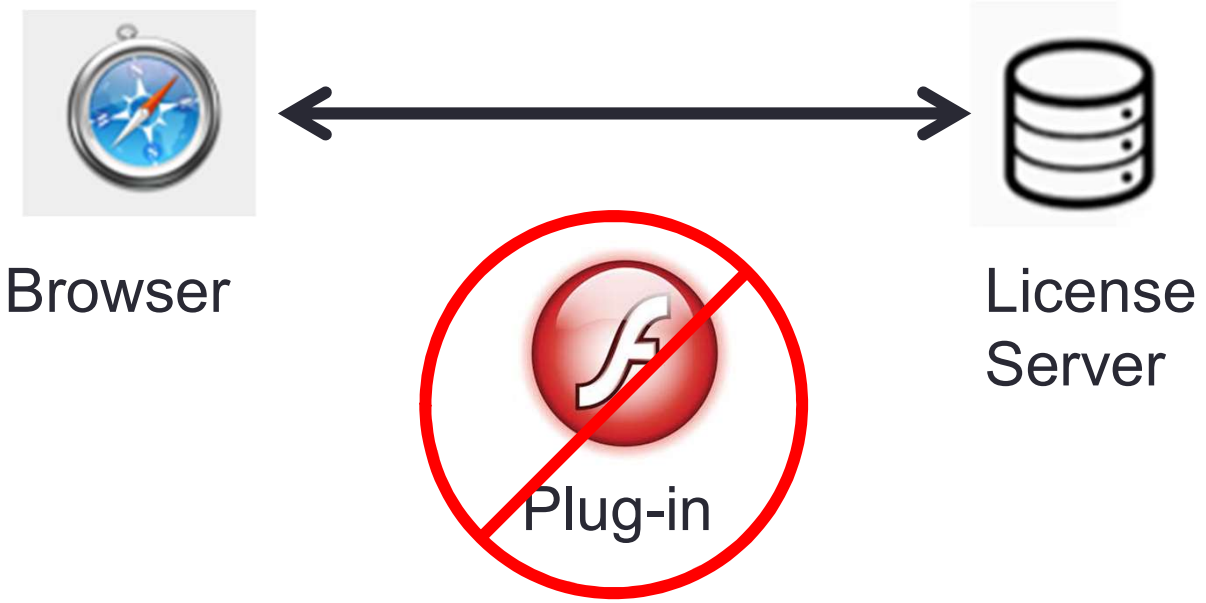
From Plug-ins to EME

- Previously – most DRM is implemented via plugins or Apps
  - Browser essentially hands all DRM and playback-related functions to the player or plug-in



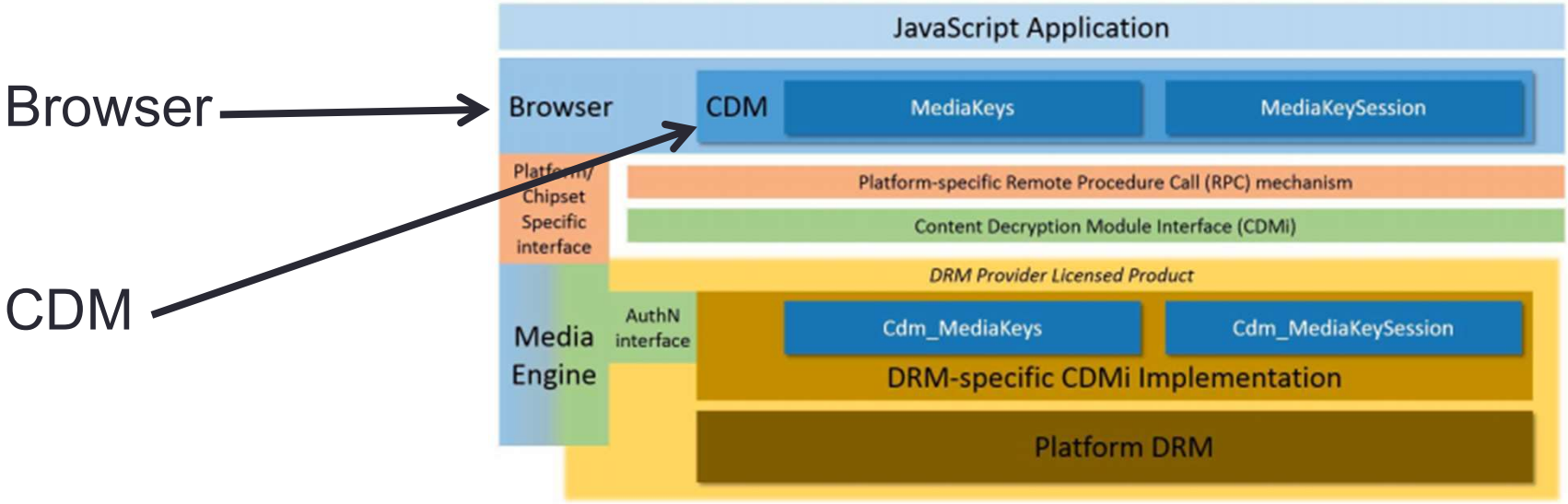
From Plug-ins to EME

- Encrypted Media Extension – DRM is baked into the browser
  - Browser handles all DRM and playback-related functions



EME uses concept of Content Decryption Modules

- Content Decryption Module – The DRM component formerly in the player/plug-in
  - Baked into browser within CDM interface



From Content Decryption Module Interface Spec http://bit.ly/DRM_EME3

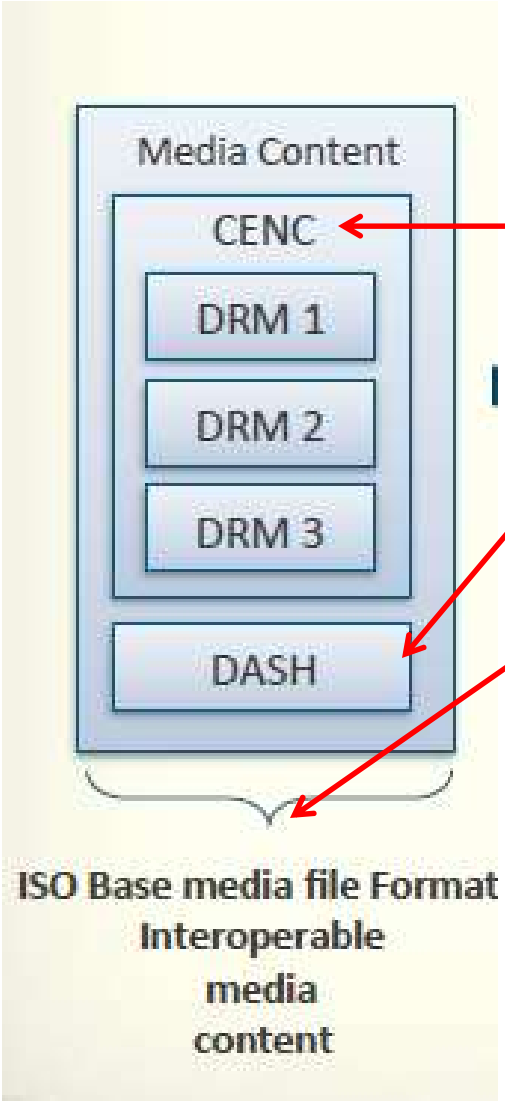
What is Common Encryption?

- Standardized encryption and key methods
  - Includes simple clear Key encryption (like HLS AES encryption)
  - AND, multiple true DRMs (with license servers) to be applied to same file
    - We'll see why this is important in a moment

What is ISO Base Media File Format (ISO-BMFF)?

- This is the standard file format used to distribute:
  - DASH encoded video
  - CENC DRM information
- Let’s see what it looks like

How DASH/EME/ISO-BMFF Fit together



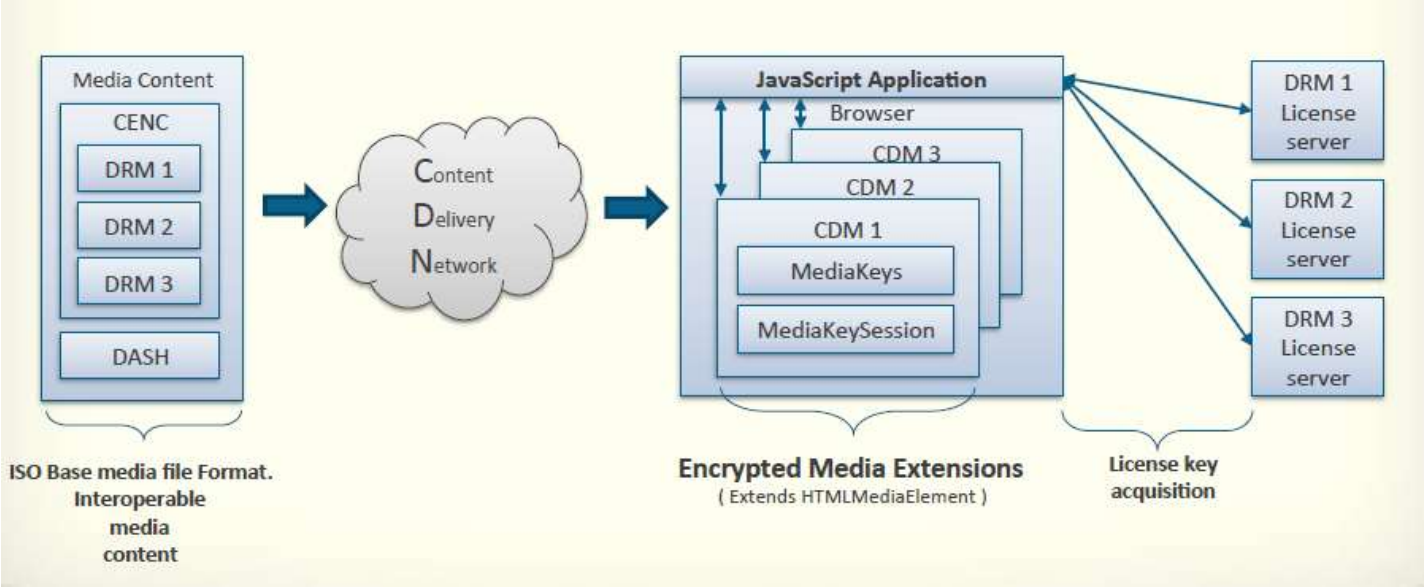
- CENC contains standardized DRM info
- DASH is compressed audio/video
- Stored in ISO-BMFF file

Image courtesy BuyDRM

The Big Picture

DASH – File format

EME – native playback



- Encrypted Media Extensions parses file
- Communicates with appropriate license server

- EME gets key and decrypts content
- MSE plays the file

Image courtesy BuyDRM

MSE/EME and Browser Share

- Support not universal (MSE more than EME)
- Notable holdouts
  - IE 11 – EME for Windows 8 only
  - Older versions of IE are neither MSE/EME compatible
- Need to check logs to determine % of users covered
- May need fallback to Flash for older browsers

Implementing EME

- The problem with EME
- Potential workarounds

The Problem Is

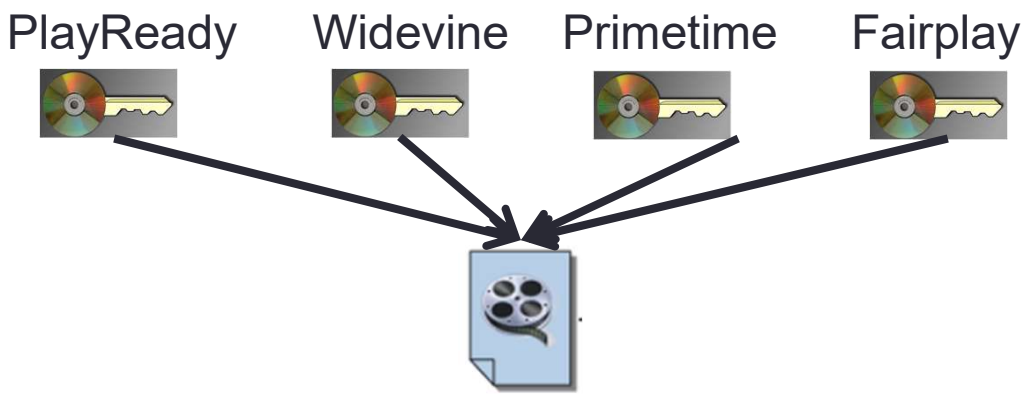
HTML5 Browsers	PlayReady	Widevine MODULAR	Widevine CLASSIC	FairPlay	Primetime (ACCESS)
Chrome (35+)	✗	✓	✗	✗	✗
Firefox (38+ on Windows)	✗	✗	✗	✗	✓
Firefox (47+ on Windows & Mac) [1]	✗	✓	✗	✗	✓
Internet Explorer (11+ on Windows 8.1+)	✓	✗	✗	✗	✗
Microsoft Edge (Windows 10+)	✓	✗	✗	✗	✗
Opera (31+)	✗	✓	✗	✗	✗
Safari (8+ on OS X)	✗	✗	✗	✓	✗

https://drmtoday.com/platforms/

- All browsers/platforms support one or two CDM
  - MS browser and mobile – PlayReady
  - Google browser, Android and devices – Widevine
  - Apple – FairPlay
  - Firefox – Primetime/Widevine
- So, need multiple DRMs to reach all targets

It's OK from a File Creation Standpoint

- Using MPEG DASH (a media format) plus CENC (Common Encryption Scheme), a single file (or adaptive group of files) can *contain multiple DRM key technologies*



But to Achieve Multiple Platform Support

- You'll need to separately support all DRMs on all platforms that you're hoping to support


Widevine


Primetime Widevine


FairPlay


PlayReady


PlayReady (IE11/Win8

FairPlay


Widevine

- To access the platforms you can access via one technology today, you'll need to support four

Multi-DRM Service Providers

- Adobe Primetime DRM
- Azure
- BuyDRM
- Cisco VideoGuard Everywhere
- DRM Today
- EZDRM
- ExpressPlay
- Verimatrix
- Vualto DRM

Common Media Application Format

- CMAF
  - Apple announcement June 2016
  - HLS can use fMP4 files rather than .ts
  - But, two incompatible encryption schemes
    - CBC (Cipher Block Chaining-Apple) and CTR (Counter Mode-everyone else)
    - Still need two copies of content
  - Since then
    - Google suppoorts CBC in Widevine
    - Microsoft supposedly supporting in PlayReady, but no formal announcement

Before CMAF

DASH

HLS

After CMAF

CMAF/CBC

CMAF/CTR

Distributing to Mobile

- Producing for iOS
  - Adaptive technologies
  - Apple recommendations
- Producing for Android
  - Adaptive technologies
  - Google's recommendations
- Encoding approaches
  - Single group/transmux
  - Separate group for each target

Encoding for iOS

- Native support
  - HTTP Live Streaming
- Via App
  - Any, including DASH, Smooth, HDS or RTMP Dynamic Streaming
  - Some risk of App store rejection (but considered very minor)

- Technical Q&A QA1767
- Your application will be rejected with the 9.4 notice if:
  - You are not using HTTP Live Streaming when streaming video longer than 10 minutes in duration over a cellular network connection.

Encoding for iOS Devices – H.264

HDR (HEVC) 30 fps	HEVC/H.265 30 fps	H.264/AVC	Resolution 16:9 aspect ratio	Frame rate
160	145	145	416 x 234	≤ 30 fps
360	300	365	480 x 270	≤ 30 fps
800	660	730	640 x 360	≤ 30 fps
1200	990	1100	768 x 432	≤ 30 fps
2050	1700	2000	960 x 540	same as source
2900	2400	3000	1280 x 720	same as source
3850	3200	4500	1280 x 720	same as source
5400	4500	6000	1920 x 1080	same as source
7000	5800	7800	1920 x 1080	same as source
9700	8100	n/a	2560 x 1440	same as source
13900	11600	n/a	3840 x 2160	same as source
20000	16800	n/a	3840 x 2160	same as source

- Significant change:
  - Expect all to play High Profile
  - Keyframe – 2 seconds
  - Segment size – 6 seconds
  - Still 200% constrained VBR

http://bit.ly/A_Devices_Spec

Encoding for iOS Devices – HEVC

HDR (HEVC) 30 fps	HEVC/H.265 30 fps	H.264/AVC	Resolution 16:9 aspect ratio	Frame rate
160	145	145	416 x 234	≤ 30 fps
360	300	365	480 x 270	≤ 30 fps
800	660	730	640 x 360	≤ 30 fps
1200	990	1100	768 x 432	≤ 30 fps
2050	1700	2000	960 x 540	same as source
2900	2400	3000	1280 x 720	same as source
3850	3200	4500	1280 x 720	same as source
5400	4500	6000	1920 x 1080	same as source
7000	5800	7800	1920 x 1080	same as source
9700	8100	n/a	2560 x 1440	same as source
13900	11600	n/a	3840 x 2160	same as source
20000	16800	n/a	3840 x 2160	same as source

- WWDC – June 2017
  - Max is Main 10, Level 5
  - Must be fMP4
  - Should provide H.264 for backwards compatibility
  - Should provide quick play

http://bit.ly/A_Devices_Spec

Encoding for iOS Devices – HDR

HDR (HEVC) 30 fps	HEVC/H.265 30 fps	H.264/AVC	Resolution 16:9 aspect ratio	Frame rate
160	145	145	416 x 234	≤ 30 fps
360	300	365	480 x 270	≤ 30 fps
800	660	730	640 x 360	≤ 30 fps
1200	990	1100	768 x 432	≤ 30 fps
2050	1700	2000	960 x 540	same as source
2900	2400	3000	1280 x 720	same as source
3850	3200	4500	1280 x 720	same as source
5400	4500	6000	1920 x 1080	same as source
7000	5800	7800	1920 x 1080	same as source
9700	8100	n/a	2560 x 1440	same as source
13900	11600	n/a	3840 x 2160	same as source
20000	16800	n/a	3840 x 2160	same as source

- WWDC – June 2017
  - Must be HDR10/Dolby
  - 30 fps or less
  - Must be fMP4
  - If HDR, must provide in all resolutions
  - Must provide SDR
  - Must provide H.264
  - Should provide quick play streams

http://bit.ly/A_Devices_Spec

Encoding for Android – Adaptive Technologies

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.6%
4.1.x	Jelly Bean	16	2.3%
4.2.x		17	3.3%
4.3		18	1.0%
4.4	KitKat	19	14.5%
5.0	Lollipop	21	6.7%
5.1		22	21.0%
6.0	Marshmallow	23	32.0%
7.0	Nougat	24	15.8%
7.1		25	2.0%
8.0	Oreo	26	0.2%

http://bit.ly/And_ver

Codecs	ABR
VP8 (2.3+) ↓	
H.264 (3+) ↓	HLS (3+) ↓
VP9 (4.4+) ↓	DASH 4.4+ Via MSE in Chrome ↓
HEVC (5+) ↓	

http://bit.ly/androidvideospecs

- Multiple codecs and ABR technologies
  - Serious cautions about HLS
  - DASH now close to 93%

Encoding for Android Devices

- Android support is bifurcated
  - In OS software – Baseline profile only
  - In hardware/device supplied software, up to High
  - Google recommends using Baseline (bit.ly/androidvideospecs)
    - Ignored by many

Table 2. Examples of supported video encoding parameters for the H.264 Baseline Profile codec.

	SD (Low quality)	SD (High quality)	HD 720p (N/A on all devices)
Video resolution	176 x 144 px	480 x 360 px	1280 x 720 px
Video frame rate	12 fps	30 fps	30 fps
Video bitrate	56 Kbps	500 Kbps	2 Mbps
Audio codec	AAC-LC	AAC-LC	AAC-LC
Audio channels	1 (mono)	2 (stereo)	2 (stereo)
Audio bitrate	24 Kbps	128 Kbps	192 Kbps

Encoding for Mobile - Choices

- Ignore older devices – all high profile
- One set of files – mixed baseline, main, high, for all targets
  - Cheapest, easiest
  - May be leaving some quality on the table
- Separate ABR groups customized for devices:
  - Baseline – old iOS and Android
  - Main – old iOS and Android
  - High – new iOS, computers and OTT
  - Optimal quality, but more encoding, storage and administrative costs

How Much Quality Difference?

Talking Head	Data Rate	Baseline	High	Delta
234p	145,000	33.79	34.20	1.22%
270p	350,000	35.72	35.99	0.75%
360p	600,000	38.16	38.37	0.54%
540p	1,000,000	40.04	40.33	0.71%
720p	1,500,000	40.78	41.32	1.34%
1080p	2,500,000	43.53	44.11	1.34%
Average		**38.67**	**39.05**	**0.98%**

Haunted	Data Rate	Baseline	High	Delta
234p	145,000	30.46	31.56	3.61%
270p	365,000	33.14	33.73	1.79%
360p	900,000	35.99	36.38	1.10%
540p	1,500,000	38.09	38.62	1.38%
720p	2,500,000	39.28	39.84	1.42%
1080p	6,000,000	41.31	41.86	1.32%
Average		**36.38**	**37.00**	**1.77%**

- Talking head on left, DSLR movie footage on right
- FFmpeg/x264/New TN2224/PSNR
- Very minor difference at all configurations

Conclusions

- Decision is content dependent
  - Challenging footage, perhaps worthwhile
  - Talking head, almost certainly not
- Even with challenging footage, Main delivers very close to the same quality as High
  - So it may never make sense to re-encode Main encoded streams to High

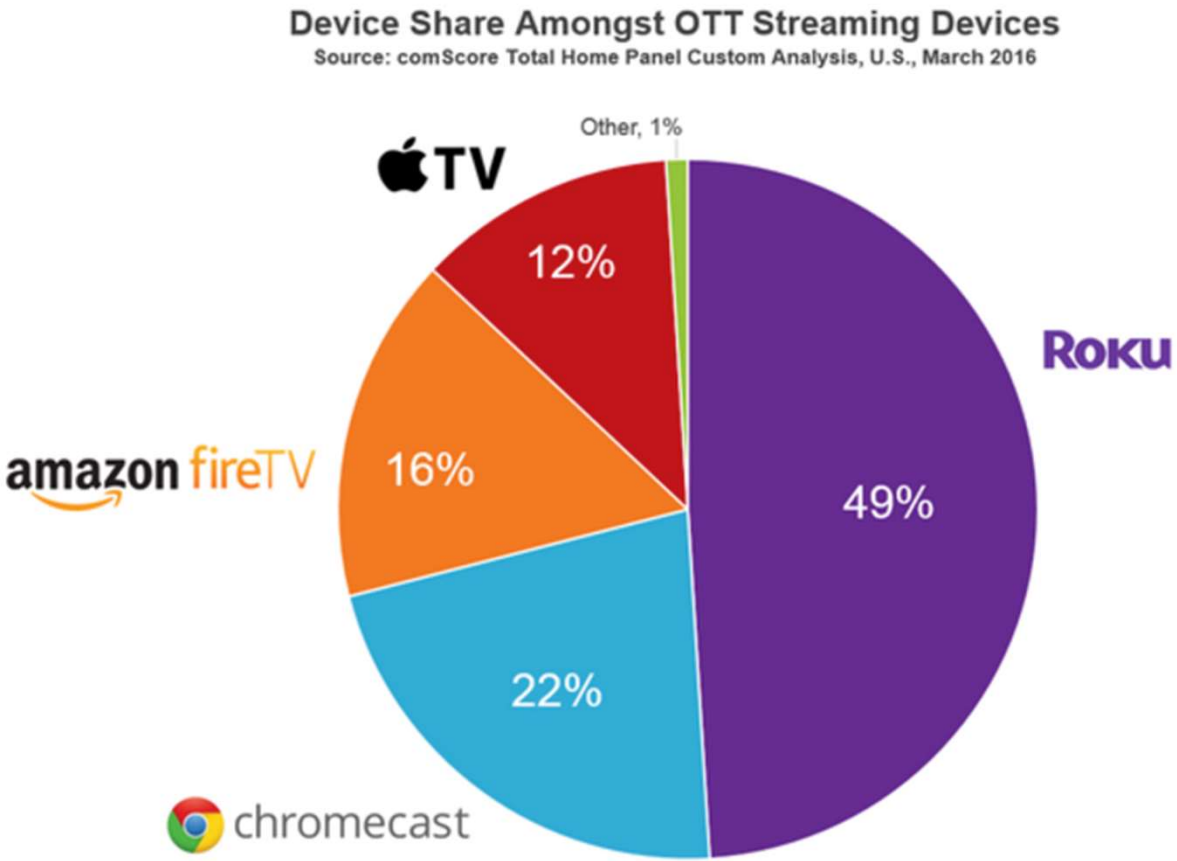
Conclusions

- Overall, if you have to create separate adaptive groups for other reasons (DRM, captions, format support), customize by device
  - High for computers/OTT
  - Baseline where necessary for mobile
- If you don’t, than it may not be worthwhile
  - Test your footage and see

Adaptive Streaming to OTT

- Format support – general
- Roku
- Apple TV
- Chromecast
- Amazon Fire TV
- PS3/PS4
- Xbox 360/Xbox One

Who Matters?



http://bit.ly/mar_16_ott

OTT Platform-Format Support

Platform	Smooth Streaming	HLS	DASH
OTT Platforms			
Roku (bit.ly/encode_roku)	Yes	Yes	Yes
Apple TV (bit.ly/AppleTV_recs)	No	Yes	No
ChromeCast (bit.ly/Chromecast_media)	Yes	Yes	Yes
Amazon Fire TV (bit.ly/Firetv_media)	Yes	Yes	Yes

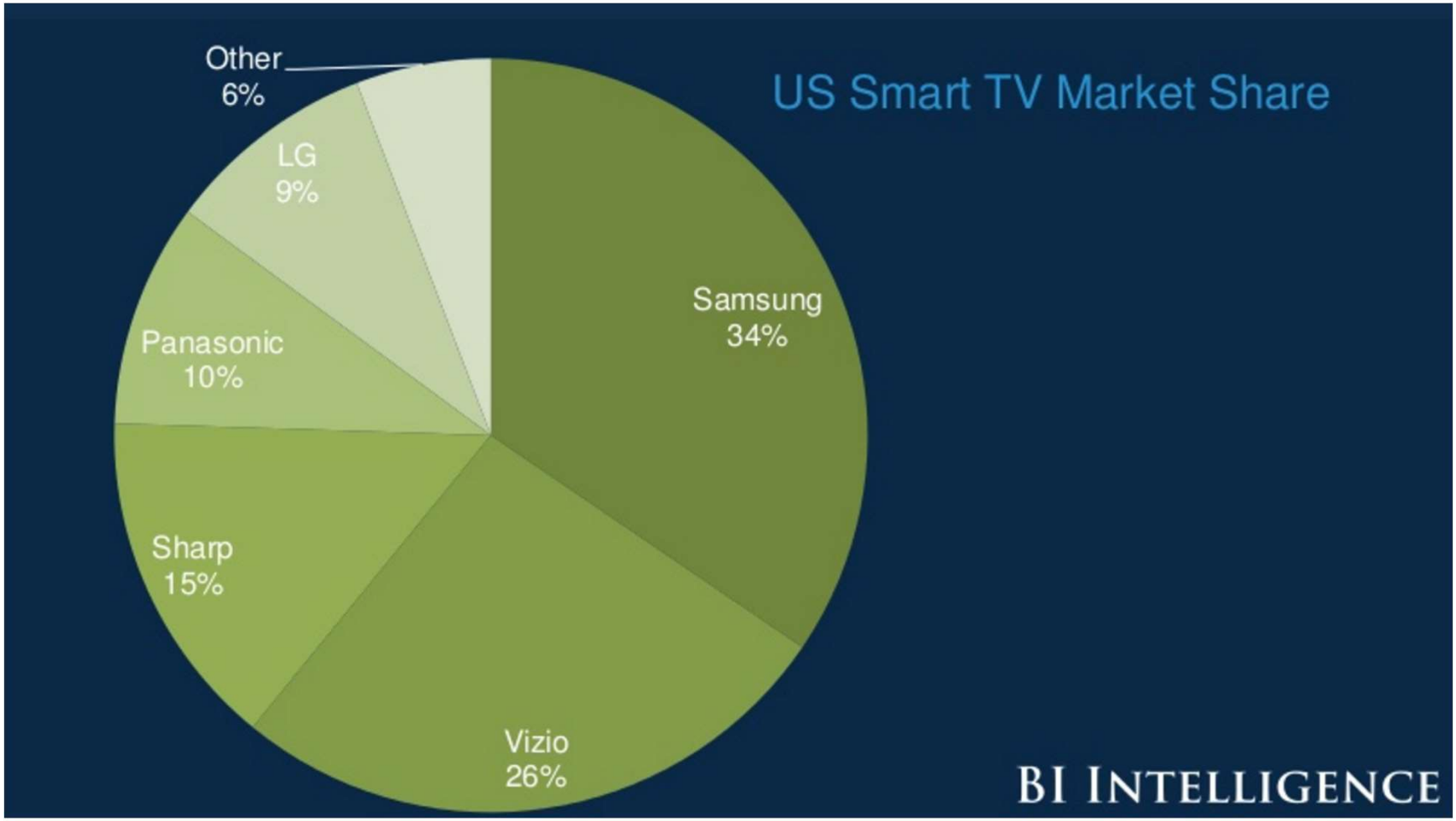
Notes:

- Roku 4 and Roku4 TVs supports HEVC and VP9
- Fire TV Gen 2 supports HEVC
- Fire TV Supports VP9
- Most recent Apple TV specs do support CMAF

Adaptive Streaming to Smart TVs

- Format support – general
- Samsung
- Vizio
- Sharp
- Panasonic
- LG
- Smart TV Alliance
- HbbTV

Who Matters?



http://www.slideshare.net/bi_intelligence/bii-the-internet-of-everything-2015

Samsung Format Support

- Samsung changed platforms in 2015 to Tizen
  - Old specs - bit.ly/Samsung_oldspec
  - Tizen – spec - bit.ly/Tizen_spec
  - On Tizen, features differ for native or SDK

	Tizen TV	SDK 2.0
codecs	HEVC, H.264, VP8, VP9	
Streaming formats	MPEG-DASH, HLS, Smooth	MPEG-DASH, HLS, Smooth
DRM	PlayReady, Widevine, AES128, Verimatrix, SecureMedia	
Captions	SMPTE-TT, DFXP	SMPTE-TT, DFXP

Vizio Format Support - ?

- Codecs
- ABR formats
- DRM
- Captions

Sharp Format Support -?

- Codecs
- ABR formats
- DRM
- Captions

Smart TV Alliance

- Members
  - Panasonic, LG, Toshiba
- Spec – 5.0 (9/2015)
- Codecs
  - H.264, HEVC
- ABR formats
  - MPEG DASH, Smooth Streaming, HLS
- DRM
  - PlayReady, Widevine
- Captions
  - W3C TTML

bit.ly/STA_spec

HbbTV 2.01 – 4/16/2016

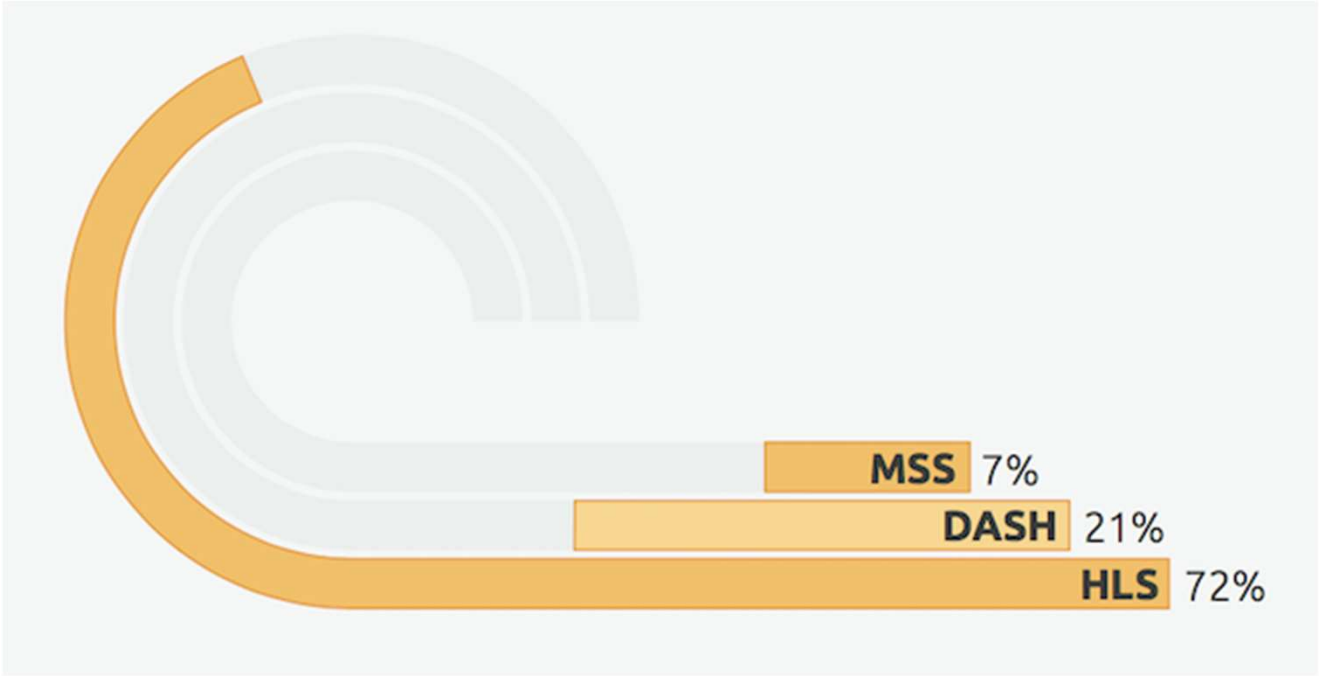
- Codecs
  - H.264, HEVC
- ABR formats
  - DASH
- DRM
  - CENC
- Captions
  - W3C TTML

bit.ly/HbbTV_201

ABR Market Share – 2015

- encoding.com 2016 Global Media Format Report
- Smooth Streaming – largely Xbox
- http://bit.ly/ecmf2016

ABR Market Share – 2016



- encoding.com 2016 http://bit.ly/ecmf_2017
- HLS – gained 1% point
- DASH – gained 11% (before royalty situation arose)
- Smooth lost 12%

Synthesis

- DASH royalty changes everything
  - Stick with HLS as long as possible and use as much as possible
- Best approach is to choose a player/encoding/delivery schema that can ***quickly and affordably address both***
  - Dynamic packaging
  - Two step encoding

Targeting Your Platforms

- Questions
- Should be 11:00 - ish

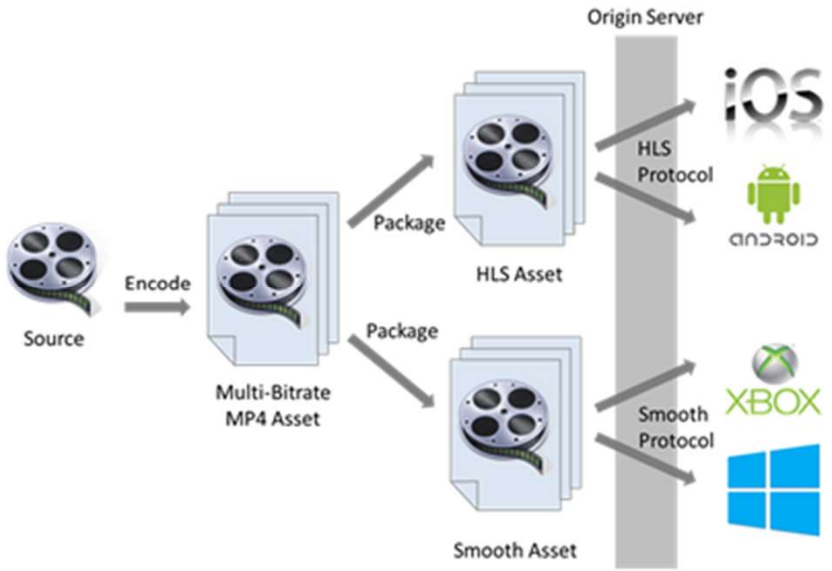
Encoding and Delivering

- Static vs. dynamic delivery
  - Defined
  - Choosing between the two
- Encoding for static delivery
  - Existing workflow
  - Encoding then packaging
  - Tool options
- Dynamic delivery
  - DIY options
  - Third party service providers

Static vs. Dynamic Delivery

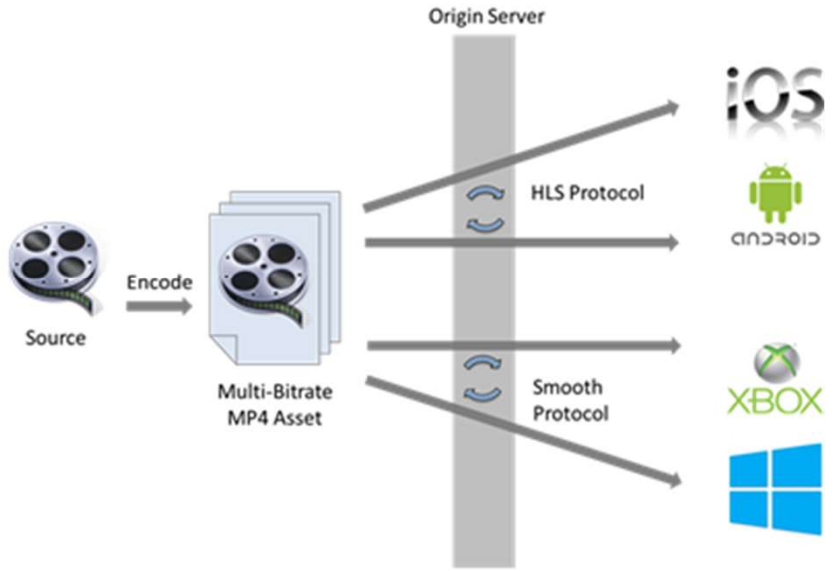
Static

- Create multi-bitrate MP4 files from mezz file
- Create ABR files from multi-bitrate files
- Upload ABR files to server
- Distribute ABR files from origin server



Dynamic

- Create multi-bitrate MP4 files and store on server
- Server dynamically creates ABR chunks and manifest files as needed
  - Via product: Wowza, Elemental, Kaltura NGINX-VOD-Module, Nimble Streamer
  - Via service: Azure, Limelight, Akamai



Static vs. Dynamic Delivery

Static: Pros/Cons

- Pros
  - Simple, no streaming server required
- Cons
  - Storage intensive
  - Major effort to support new formats
    - Must create new packaged files
    - Upload to servers

Dynamic: Pros/Cons

- Pros
  - Storage efficient
  - Very simple to support new formats/devices down the road
- Cons
  - More technically complex
  - May be more expensive
    - If server component costs more than extra storage

Static vs. Dynamic

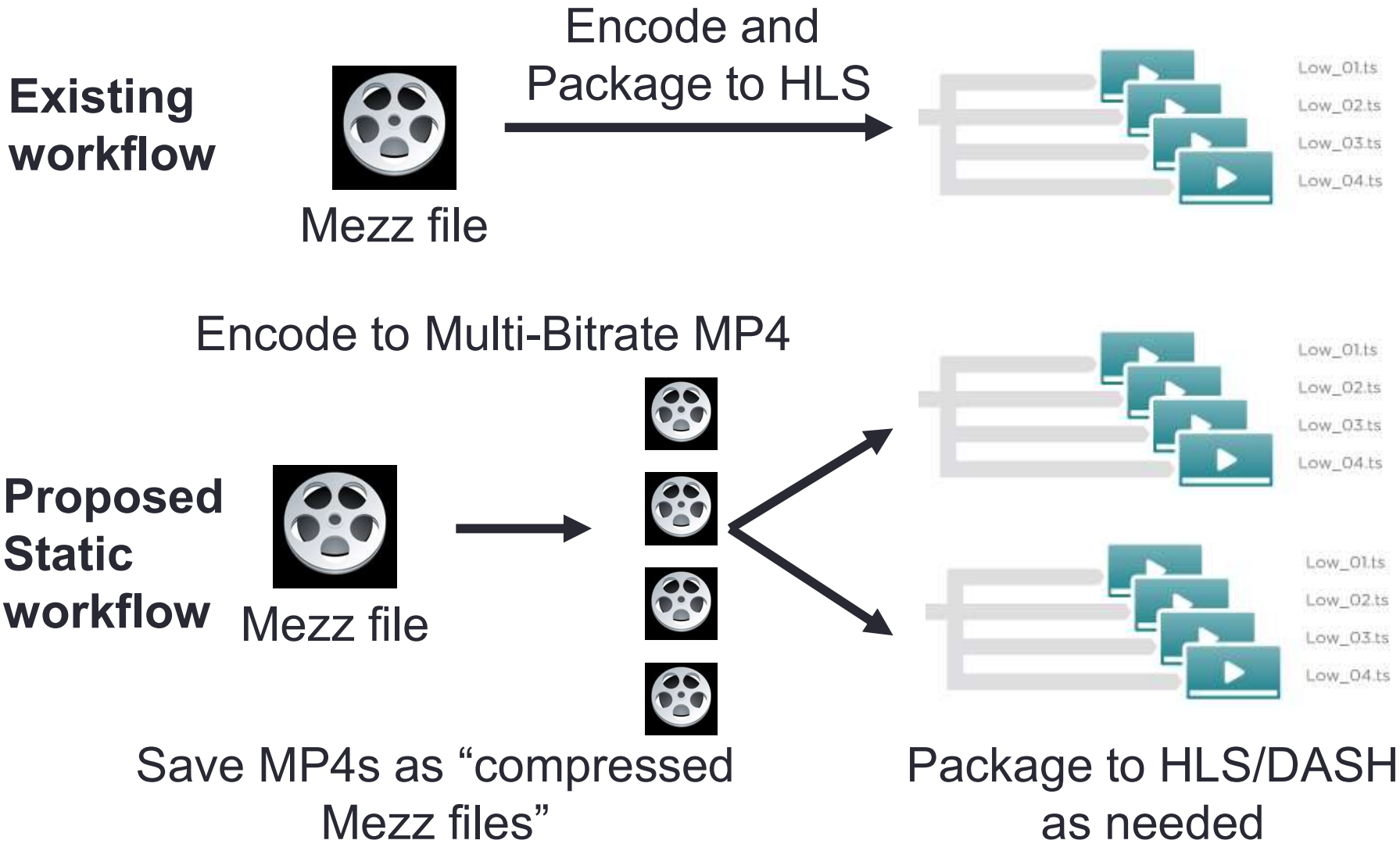
- Consulting project; cloud encoding for library and ongoing
  - Static – increased encoding and storage costs
  - Dynamic – increased server (Wowza), but much cheaper

Two Year Projections	Static ABR/ Encode Library ASAP	Dynamic ABR - encode library ASAP	Delta
Ongoing cloud encode	$68,651	$49,914	-$18,737
Wowza		$20,405	$20,405
Extra storage for HLS/DASH	$34,009		-$34,009
Total ongoing	**$102,660**	**$70,319**	**-$32,341**
One Time Library Conversion	$125,091	$55,114	-$69,977
Total	**$227,751**	**$125,433**	**-$102,318**

Updating the Static File Creation Workflow

- If static selected, need to update encoding workflow to leverage similar benefits
  - Typical existing ***single-step*** workflow
    - Encode from mezz to final ABR formats
    - Complete re-encode needed to support new formats (like HLS > DASH migration)
  - More efficient ***two-step workflow***
    - ***Encode step*** - Encode from mez to MP4 (use as compressed mezz files)
    - ***Package step*** – transmux multi-bitrate MP4 files into ABR formats

Updating The Static Encoding Workflow



Updating the Static File Creation Workflow

- **Encode step** – encode mezz files into multi-bitrate MP4s:
  - Used as source for ABR files
  - This is the expensive, time-consuming step
  - Won't change when it's time to support new ABR formats
- **Package step** – transmux multi-bitrate MP4 files into ABR formats
  - This is fast and cheap
  - Easy to support new formats like DASH

Choosing an Encoder/Packager

Encoder

- Any desktop, enterprise or cloud encoder that can create MP4 files

DASH Packagers

- edash-packager
  - bit.ly/Dash_pack1
- MP4Box - http://gpac.io.
- Rebaca MPEG DASH Segmenter
  - http://bit.ly/Dash_pack2
- castLabs Dash Encrypt Packager
  - https://github.com/castlabs/dashencrypt
- Bento4 - www.bento4.com

Choosing an Encoder/Packager

HLS Packagers

- Apple Media Stream Segmenter (MPEG-2 transport streams)
- Apple Media File Segmenter (MP4 inputs)
  - http://bit.ly/HLS_pack
- Apple Variant Playlist Creator
- FFmpeg – media playlists and packaging only
  - No master m3u8
- Bento4

Other Packagers

- Unified Packager (DASH, HLS, HDS, Smooth)
  - bit.ly/Uni_pack
- ProMedia Package (HLS, Smooth, HDS, DASH)
  - bit.ly/harm_pack

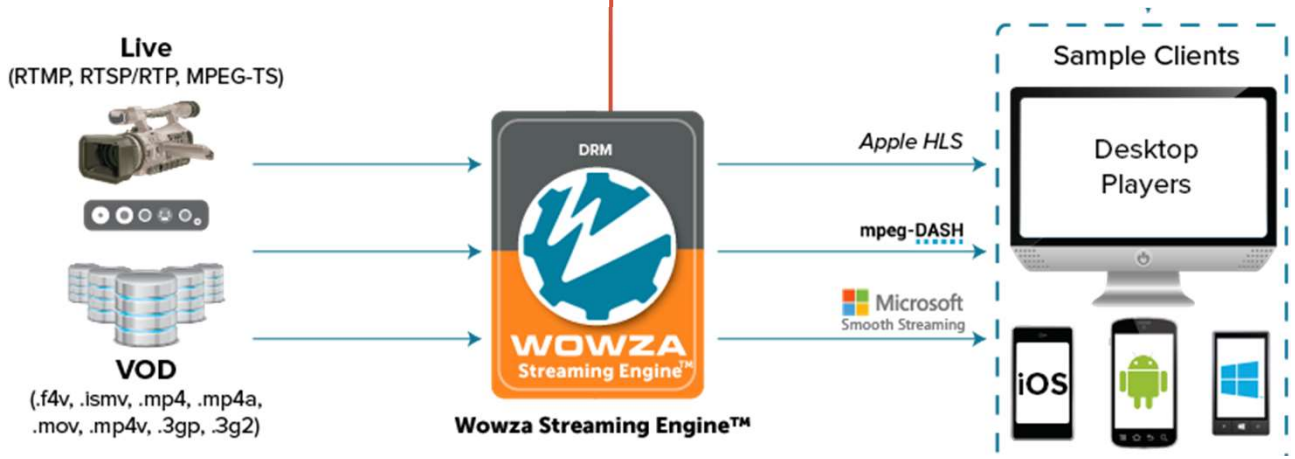
Dynamic Alternatives

DIY

- Wowza Streaming Engine
- Nimble Streamer
- Elemental Delta
- Azure Media Services
- Kaltura tool
- Telurio Packager

Service Providers

- Akamai
- Limelight



Packaging and Delivering

- Questions?
- Should be 11:15 - ish

Configuring Adaptive Streams

- Basic configuration options
  - Number of streams
  - Resolutions/data rates
  - H.264 parameters
- Encoding adaptive streams
  - Key frame settings
  - Bitrate control
  - Audio settings

The Big Question

- One set of streams for all devices?
- Customize streams for each class

Number of Streams

- There is no right or wrong way
- I recommend:
  - Attempt to find one set of streams that satisfies all using some layers encoded with Baseline profile
  - Test to see if Main or High delivers noticeably higher quality
    - If so, customize for different targets
    - If not, go with one set of streams

Configuring Your Streams: Mobile First

- Slowest connection, lowest quality
  - Extra rules apply for app store approval
- Configure for best service to mobile
- Try to configure at same resolutions as low end computer targets

16:9 Aspect Ratio	Dimensions	Frame Rate *	Total Bit Rate	Video Bit Rate	Audio Bit Rate	Audio Sample Rate	Keyframe**	Restrict Profile to:	Works on iPod Touch Gens 2, 3, 4	Works on iPhone 3G, 3GS, 4	Works on iPad 1, 2	Works on New iPad	Works Apple T
CELL	480x320	na	64	na	64	44.1	na	na	*	*	*	*	*
CELL	416x234	10 to 12	264	200	64	44.1	30 to 36	Baseline, 3.0	*	*	*	*	*
CELL	480x270	12 to 15	464	400	64	44.1	36 to 45	Baseline, 3.0	*	*	*	*	*
WIFI	640x360	29.97	664	600	64	44.1	90	Baseline, 3.0	*	*	*	*	*
WIFI	640x360	29.97	1264	1200	64	44.1	90	Baseline, 3.1			*	*	*
WIFI	960x540	29.97	1864	1800	64	44.1	90	Main, 3.1			*	*	*
WIFI	960x540	29.97	2564	2500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1280x720	29.97	4564	4500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1280x720	29.97	6564	6500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1920x1080	29.97	8564	8500	64	44.1	90	High, 4.0				*	*

Apple Tech Note TN2224

Desktop (browser-based) Next

- At least one stream for each window size in web site (HBO)
- Try to use same configuration as mobile

Scenario	Format	Frame Size	Total Bitrate	Audio Bitrate	bits/pixel *frame @ 30 fps	bits/pixel *frame @ 24 fps
Mobile & constrained (low)	baseline, mono, 10 fps	448x252	150	48	0.09	0.09
Mobile & constrained (high)	baseline, mono	448x252	450	48	0.12	0.15
Sidebar placements	main profile, stereo	384x216	400	96	0.12	0.15
Small in-page	main profile, stereo	512x288	750	96	0.15	0.18
Medium in-page	main profile, stereo	640x360	1200	96	0.16	0.20
Large in-page	main profile, stereo	768x432	1700	96	0.16	0.20
Full size in-page	main profile, stereo	960x540	2200	96	0.14	0.17
HD 720p (full screen)	high profile, stereo	1280x720	3500	96	0.12	0.15

Then High End Mobile, Desktop and OTT

- Full screen viewing on all devices
- Highest quality streams that you can afford

16:9 Aspect Ratio	Dimensions	Frame Rate *	Total Bit Rate	Video Bit Rate	Audio Bit Rate	Audio Sample Rate	Keyframe**	Restrict Profile to:	Works on iPod Touch Gens 2, 3, 4	Works on iPhone 3G, 3GS, 4	Works on iPad 1, 2	Works on New iPad	Works on Apple TV 2
CELL	480x320	na	64	na	64	44.1	na	na	*	*	*	*	*
CELL	416x234	10 to 12	264	200	64	44.1	30 to 36	Baseline, 3.0	*	*	*	*	*
CELL	480x270	12 to 15	464	400	64	44.1	36 to 45	Baseline, 3.0	*	*	*	*	*
WIFI	640x360	29.97	664	600	64	44.1	90	Baseline, 3.0	*	*	*	*	*
WIFI	640x360	29.97	1264	1200	64	44.1	90	Baseline, 3.1			*	*	*
WIFI	960x540	29.97	1864	1800	64	44.1	90	Main, 3.1			*	*	*
WIFI	960x540	29.97	2564	2500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1280x720	29.97	4564	4500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1280x720	29.97	6564	6500	64	44.1	90	Main, 3.1			*	*	*
WIFI	1920x1080	29.97	8564	8500	64	44.1	90	High, 4.0				*	*

Apple Tech Note TN2224

Stream Count – Bottom Line

- At least one stream for each playback window in website
- More streams required for HD than SD
  - SD – usually 3, 4 maximum
  - HD – up to 11
- More for entertainment than education/business
  - Entertainment – about the experience
  - Business – it's about making sure the viewer can watch the stream
- More for subscription than general entertainment
  - Provide more options when viewer is paying

What Resolution?

- Never encode at larger than source
  - Scaling upwards degrades quality
- Resolution should match playback window
  - Optimal quality and playback efficiency
- If you can't match playback window, it's better to scale up to larger resolutions than to scale down to smaller
  - More efficient playback (GPU assisted)
  - More efficient bandwidth consumption

What Data Rates?

- Apple TN2224: Keep adjacent bit rates a factor of 1.5 to 2 apart
  - If too close together, you waste band-width because quality difference is minimal (150 kbps and 180 kbps streams)
  - If too far apart, could strand some clients to lower quality stream unnecessarily

What Data Rates?

- MTV Schema

Scenario	Format	Frame Size	Total Bitrate	
Mobile & constrained (low)	baseline, mono, 10 fps	448x252	150	
Mobile & constrained (high)	baseline, mono	448x252	450	
Sidebar placements	main profile, stereo	384x216	400	350 kbps
Small in-page	main profile, stereo	512x288	750	450 kbps
Medium in-page	main profile, stereo	640x360	1200	500 kbps
Large in-page	main profile, stereo	768x432	1700	500 kbps
Full size in-page	main profile, stereo	960x540	2200	1300 kbps
HD 720p (full screen)	high profile, stereo	1280x720	3500	

How I Build Encoding Ladders

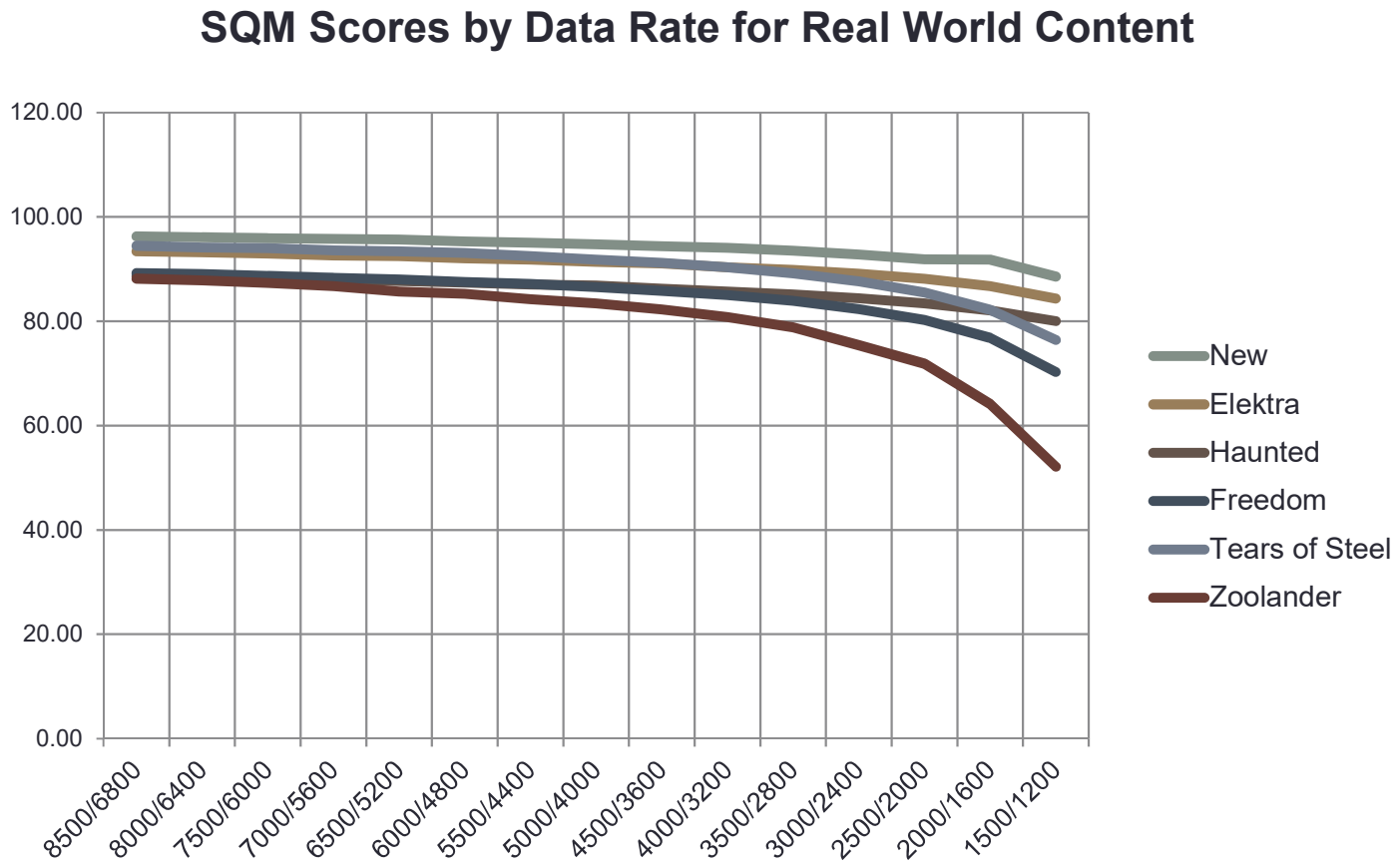
	Width	Height	Data Rate	% Jump	FPS	BPP
234p	416	234	145		15	0.099
270p	480	270	365	2.52	15	0.188
360p	640	360	730	2.00	30	0.106
432p	768	432	1100	1.51	30	0.111
540p	960	540	2000	1.82	30	0.129
720p	1280	720	3000	1.50	30	0.109
1080p_l	1920	1080	4500	1.50	30	0.072
1080p_m	1920	1080	6000	1.33	30	0.096
1080p_h	1920	1080	7800	1.30	30	0.125
1440p	2560	1440	8100	1.04	30	0.073
2160p_low	3840	2160	11600	1.43	30	0.047
2160p_high	3840	2160	16800	1.45	30	0.068

Using Objective Quality Metrics

Preset Name	Width	Height	FPS	BIT RATE	VQM	Delta	SQM	Delta
1080p_high	1920	1080	29.97	7500	0.790		95.91	
1080p_med	1920	1080	29.97	6500	0.800	-1.30%	95.77	-0.14%
1080p_low	1920	1080	29.97	5000	0.855	-6.80%	95.66	-0.12%

- Two quality metrics
  - Video Quality Metric
  - SSIMWave QOS Metric
- Insignificant difference between 6500/7500
- Very minor difference between 5000 and 6500

How Low Can You Go?



- SQM – Higher is better
- Here we see Zoolander drop below 80 right around 4 mbps
- Others stay in excellent range throughout

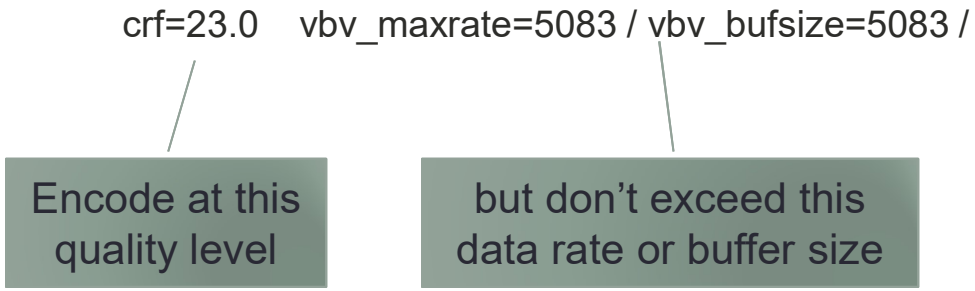
Per-Title Encoding (Content Adaptive)

- What is it?
  - Assess video file complexity and customize encoding ladder
  - For easy to encode files, saves bandwidth
  - For hard to encode files, improves quality
- Who's using it?
  - Netflix, YouTube, JWPlayer (OVP), Brightcove
- Prediction
  - In three years, all encoding will be per-title (or content adaptive)
  - If you're not thinking about this now, you should be

Per-Title Encoding – How To Do It?

- Netflix – brute force calculations
  - http://bit.ly/NF_pertitle
- YouTube
  - Neural network - http://bit.ly/yt_ca
- Capella Systems Cambria Encoder
  - Assess complexity with CRF encode
  - Adjust ladder accordingly
  - http://bit.ly/cap_cam
- JWPlayer
  - Capped CRF encoding
  - http://bit.ly/vp9_emerges
- Brightcove – context aware encoding (http://bit.ly/bc_cae)

```
# Function to get Multiplier value
sub getMultiplierValue
{
 my $complexityValue = $_[0];
 if ($complexityValue <= 0)  { return 1.0;  }
 if ($complexityValue >= 7000) {  return 1.5; }
 if ($complexityValue >= 5000) {  return 1.25; }
 if ($complexityValue >= 4000) {  return 1.0; }
 if ($complexityValue >= 3500) {  return 0.9; }
 if ($complexityValue >= 3000) {  return 0.8;  }
 if ($complexityValue >= 2500) {  return 0.7; }
 if ($complexityValue >= 2000) {  return 0.6; }
 return 0.5;
}
```



How Encoding Parameters Change for ABR

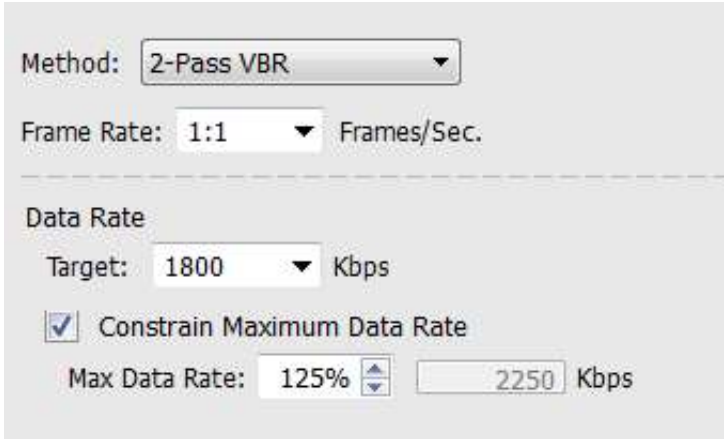
- Now we know stream count, resolution and data rate
- How do we customize encoding for adaptive?
  - Key frame settings
  - Bit rate control
  - Audio parameters

Key Frame Interval

- Why important – HTTP technologies - key frame must be first frame of every chunk
  - Key frame interval must divide evenly into chunk duration
  - If 9-sec. chunks, use key frame interval of 3 sec
- Key frame location must be identical in all streams, so:
  - Use same interval
  - Disable scene change detection

VBR vs. CBR

- Most conservative – CBR
  - Most white papers recommend this
- Or constrained VBR
  - Max data rate 110-125% of target
  - Most practitioners use 110%



In General - Audio Parameters

- Traditional approach was to use same audio parameters for all files
  - Approach used by Apple in initial versions of TN2224
  - Concern was popping that could occur when switching audio parameters
  - Now change audio parameters to match video quality

	Dimensions	Frame Rate *	Total Bit Rate	Video Bit Rate	Audio Bit Rate**
CELL	416x234	12	264	200	64
CELL	480x270	15	464	400	64
WiFi/Cell	640x360	29.97	664	600	64
WiFi	640x360	29.97	1296	1200	96
WiFi	960x540	29.97	3596	3500	96
WiFi	1280x720	29.97	5128	5000	128
WiFi	1280x720	29.97	6628	6500	128
WiFi	1920x1080	29.97	8628	8500	128

Recommended Approach

- When audio adds value (concerts, stage performances), change levels to match video quality
  - Test to make sure there are no audible artifacts
- When audio is less relevant (talking heads, etc), use a single stream for simplicity
  - 64 kbps mono

That's It

- Questions?