

ENCODING LIVE AND VOD FOR HEVC/HLS

A Joint SLC/RealEyes Production

Agenda

- Our assumptions and goals
- Section I: Introduction to HEVC
- Section II: Playback performance
- Section III: Introduction to HLS
- Section IV: Specification overview: HEVC in HLS
- Section V: Producing HEVC/HLS

Assumptions and Goals

- Assumptions
 - Have some knowledge of how to produce HLS presentations
- Goal: Teach you to *add* HEVC to HLS
 - Encode HEVC
 - Choose an HEVC encoding ladder
 - Integrate that into an HLS presentation
 - With FFmpeg, Bento4 and some third party tools
- Not a soup to nuts, here's how to do HLS session

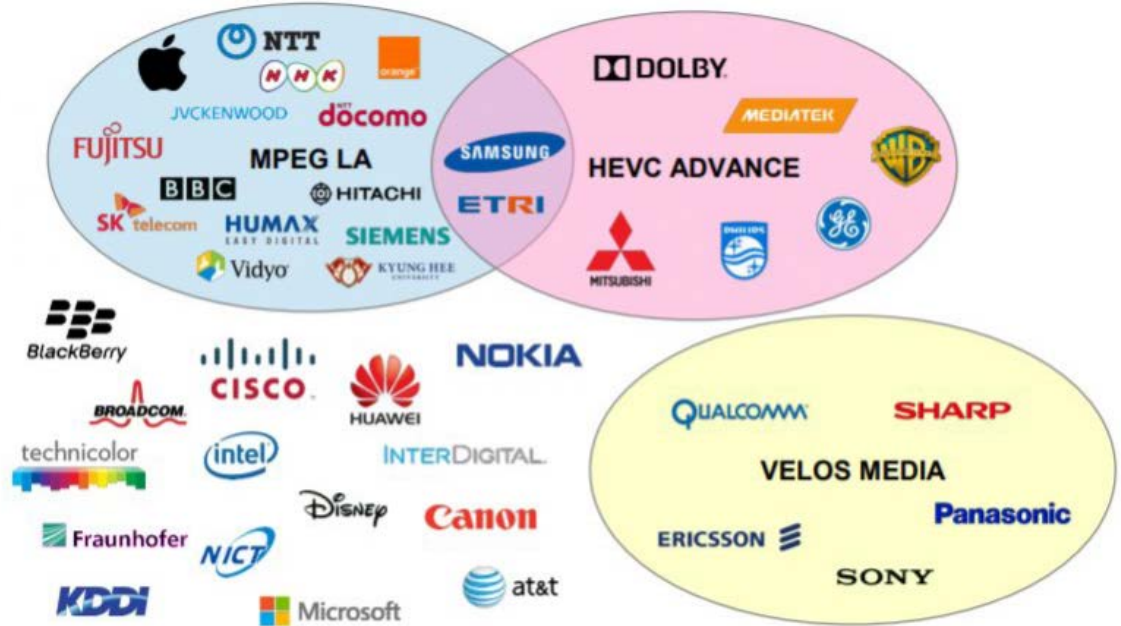
Section I. Introduction to HEVC

- About HEVC
- HEVC and royalties
- HEVC codecs
- HEVC encoding parameters
- Codec specific encoding profiles

What HEVC Is and Why It's Important

- HEVC is a standards-based compression technology
- Jointly sponsored by MPEG and ISO standards bodies
 - That's why it's called both **HEVC** and **H.265**
- OS support
 - Supported in MacOS via HLS
 - Supported in Windows 10/Edge if hardware decode is available
- Mobile – Android and iOS
- Browser support
 - MacOS/Safari, Windows 8/Edge
 - Not supported in Chrome, Firefox, Opera, or Internet Explorer

HEVC and Royalties



- Three royalty groups (MPEG-LA, HEVC Advance, and Velos)
 - MPEG-LA – no royalties on content
 - HEVC Advance – **no royalties on content**
 - Velos – may be content royalties
- Many others outside of all three

HEVC Codecs

- Because HEVC is a standard, there are many HEVC codecs
 - x265, MainConcept, Bemr, Intel, Elemental, and many others
 - x265 is the open-source HEVC encoder included with FFmpeg
 - Widely agreed to be one of highest quality HEVC codecs
 - Can also encode x265 with a separate executable, through FFmpeg is much more flexible, particularly regarding input files

Critical HEVC Encoding Parameters

- Some parameters apply to all H.265 codecs
 - Profiles
 - No matter which HEVC codec you work with, you'll have to set these
 - Levels
- Some are codec specific
 - Schema for balancing quality and encoding time

What Profiles are and Why They Exist

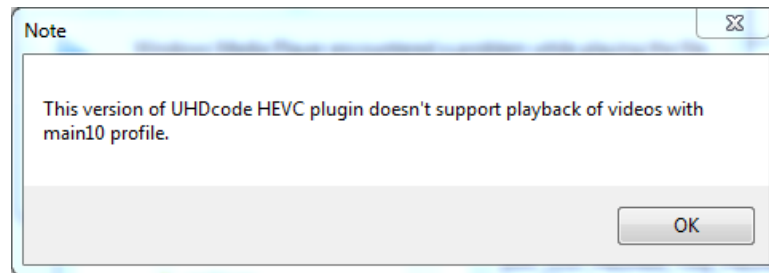
- Profiles enable different encoding techniques to balance decoding complexity
 - Version 2 codecs use more advanced features
- Apple supports both
 - 1.6. Profile, Level, and Tier for HEVC MUST be less than or equal to Main10 Profile, Level 5.0, High.

Feature	Version 1	
	Main	Main 10
Bit depth	8	8 to 10
Chroma sampling formats	4:2:0	4:2:0
4:0:0 (Monochrome)	No	No
High precision weighted prediction	No	No
Chroma QP offset list	No	No
Cross-component prediction	No	No
Intra smoothing disabling	No	No
Persistent Rice adaptation	No	No
RDPCM implicit/explicit	No	No
Transform skip block sizes larger than 4x4	No	No
Transform skip context/rotation	No	No
Extended precision processing	No	No

https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding

Main or Main10?

- Main players can't play Main 10 encoded content
 - Some early HEVC players are Main only
 - If encoding for general-purpose playback, use Main
- Main 10 has a very slight quality advantage
 - If encoding for Main10 player, use Main 10
 - Main 10 players can play Main



720p - x265	Main	Main 10	Delta
Tears of Steel	37.05	37.73	1.84%
SIintel	41.37	41.25	-0.29%
Big Buck Bunny	37.21	37.16	-0.13%
Talking Head	41.15	41.15	0.00%
Freedom	39.70	39.57	-0.31%
Haunted	39.56	41.78	5.61%
Average	39.34	39.77	1.12%

HEVC Levels

Level	Max luma sample rate (samples/s)	Max luma picture size (samples)	Max bit rate for Main and Main 10 profiles (kbit/s) ^[A]		Example picture resolution @ highest frame rate ^[B] (MaxDpbSize ^[C])
			Main tier	High tier	More/Fewer examples
1	552,960	36,864	128	–	176x144@15.0 (6)
2	3,686,400	122,880	1,500	–	352x288@30.0 (6)
2.1	7,372,800	245,760	3,000	–	640x360@30.0 (6)
3	16,588,800	552,960	6,000	–	960x540@30.0 (6)
3.1	33,177,600	983,040	10,000	–	1280x720@33.7 (6)
4	66,846,720	2,228,224	12,000	30,000	2,048x1,080@30.0 (6)
4.1	133,693,440		20,000	50,000	2,048x1,080@60.0 (6)
5	267,386,880	8,912,896	25,000	100,000	4,096x2,160@30.0 (6)

- Set constraints within profiles
- Enable compatibility with lower power devices
- Apple spec – No higher than Main10 Profile, Level 5.0, High Tier
 - That peaks at 30 fps for 4K - Apple sample streams are 60p
 - Safe to go to 60

Codec Quality/Encoding Time Presets

- Different HEVC codecs use different schemas to simplify quality/encoding time tradeoffs
 - x265 uses presets – ultra fast to placebo
 - MainConcept uses a number from 1-28
- What's important is understanding how the mechanism trades off encoding time and quality

x265 Presets

- Same name as x264; different parameters

0. ultrafast
1. superfast
2. veryfast
3. faster
4. fast
5. medium (default)
6. slow
7. slower
8. veryslow
9. placebo

preset	0	1	2	3	4	5	6	7	8	9
ctu	32	32	64	64	64	64	64	64	64	64
min-cu-size	16	8	8	8	8	8	8	8	8	8
bframes	3	3	4	4	4	4	4	8	8	8
b-adapt	0	0	0	0	0	2	2	2	2	2
rc-lookahead	5	10	15	15	15	20	25	30	40	60
lookahead-slices	8	8	8	8	8	8	4	4	1	1
scenecut	0	40	40	40	40	40	40	40	40	40
ref	1	1	2	2	3	3	4	4	5	5
limit-refs	0	0	3	3	3	3	3	2	1	0
me	dia	hex	hex	hex	hex	hex	star	star	star	star
merange	57	57	57	57	57	57	57	57	57	92
subme	0	1	1	2	2	2	3	3	4	5
rect	0	0	0	0	0	0	1	1	1	1
amp	0	0	0	0	0	0	0	1	1	1
limit-modes	0	0	0	0	0	0	1	1	1	0
max-merge	2	2	2	2	2	2	3	3	4	5
early-skip	1	1	1	1	0	0	0	0	0	0
recursion-skip	1	1	1	1	1	1	1	1	0	0
fast-intra	1	1	1	1	1	0	0	0	0	0
b-intra	0	0	0	0	0	0	0	1	1	1
sao	0	0	1	1	1	1	1	1	1	1
signhide	0	1	1	1	1	1	1	1	1	1
weightp	0	0	1	1	1	1	1	1	1	1
weightb	0	0	0	0	0	0	0	1	1	1
aq-mode	0	0	1	1	1	1	1	1	1	1
cuTree	1	1	1	1	1	1	1	1	1	1
rdLevel	2	2	2	2	2	3	4	6	6	6
rdoq-level	0	0	0	0	0	0	2	2	2	2
tu-intra	1	1	1	1	1	1	1	2	3	4
tu-inter	1	1	1	1	1	1	1	2	3	4
limit-tu	0	0	0	0	0	0	0	4	4	0

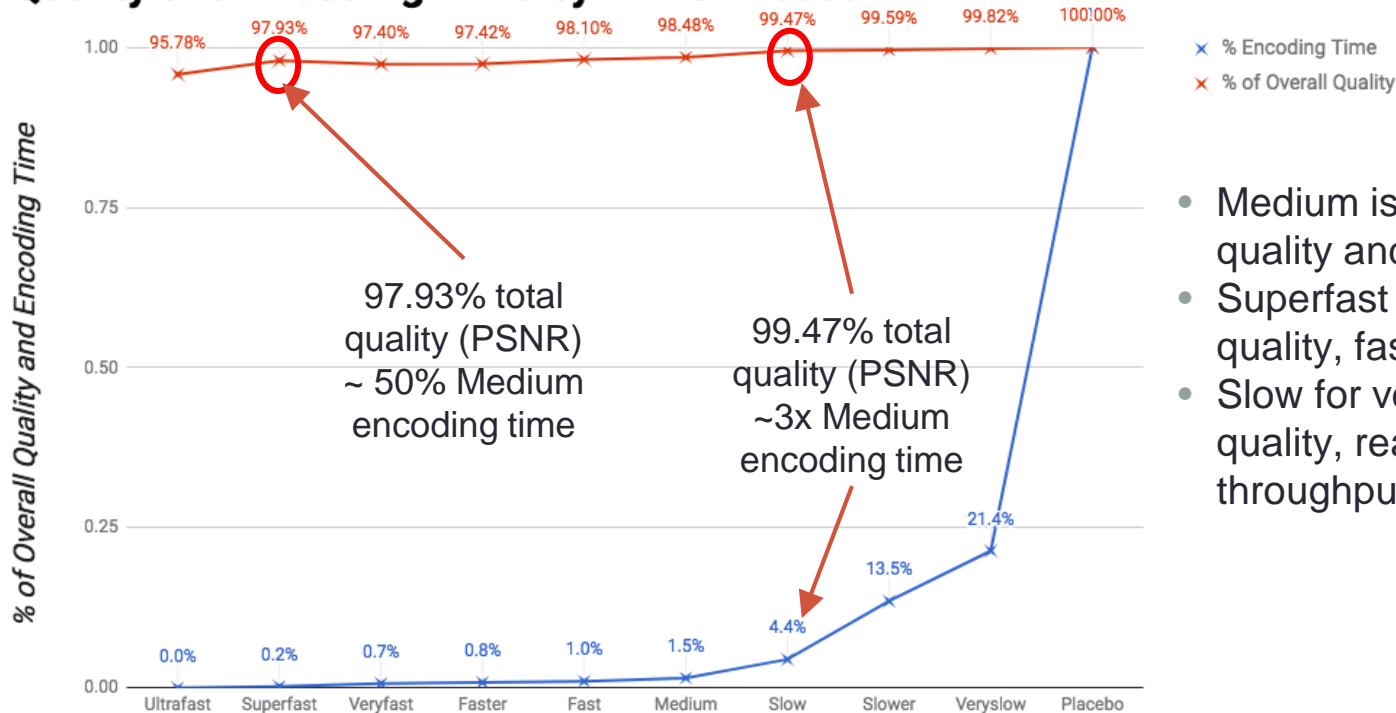
x265 Presets

	Ultrafast	Superfast	Veryfast	Faster	Fast	Medium	Slow	Slower	Veryslow	Placebo	Total Delta
Tears of Steel	37.25	38.06	38.04	38.05	38.34	38.39	38.84	38.86	38.93	39.00	4.70%
Sintel	35.87	36.89	36.66	36.67	37.11	37.25	37.74	37.79	37.90	37.97	5.86%
Big Buck Bunny	36.10	37.65	37.61	37.60	37.91	38.26	38.70	38.89	39.03	39.18	8.54%
Freedom	38.16	39.01	38.45	38.46	38.71	38.98	39.36	39.44	39.52	39.58	3.72%
Haunted	41.36	41.77	41.39	41.39	41.55	41.68	41.97	41.92	41.97	42.02	1.60%
Screencam	44.03	46.70	46.55	46.54	46.78	47.12	48.31	48.69	48.99	49.34	12.07%
Tutorial	42.46	47.14	46.46	46.42	46.52	47.19	48.35	47.65	48.02	48.53	14.31%
Average	38.64	39.51	39.30	39.31	39.58	39.74	40.13	40.18	40.27	40.35	6.70%

- Ultrafast is always the worst
 - Typically only use when necessary for live encoding
- Superfast is higher quality than Veryfast and Faster
- Starts increasingly steadily after Fast, with Placebo the best

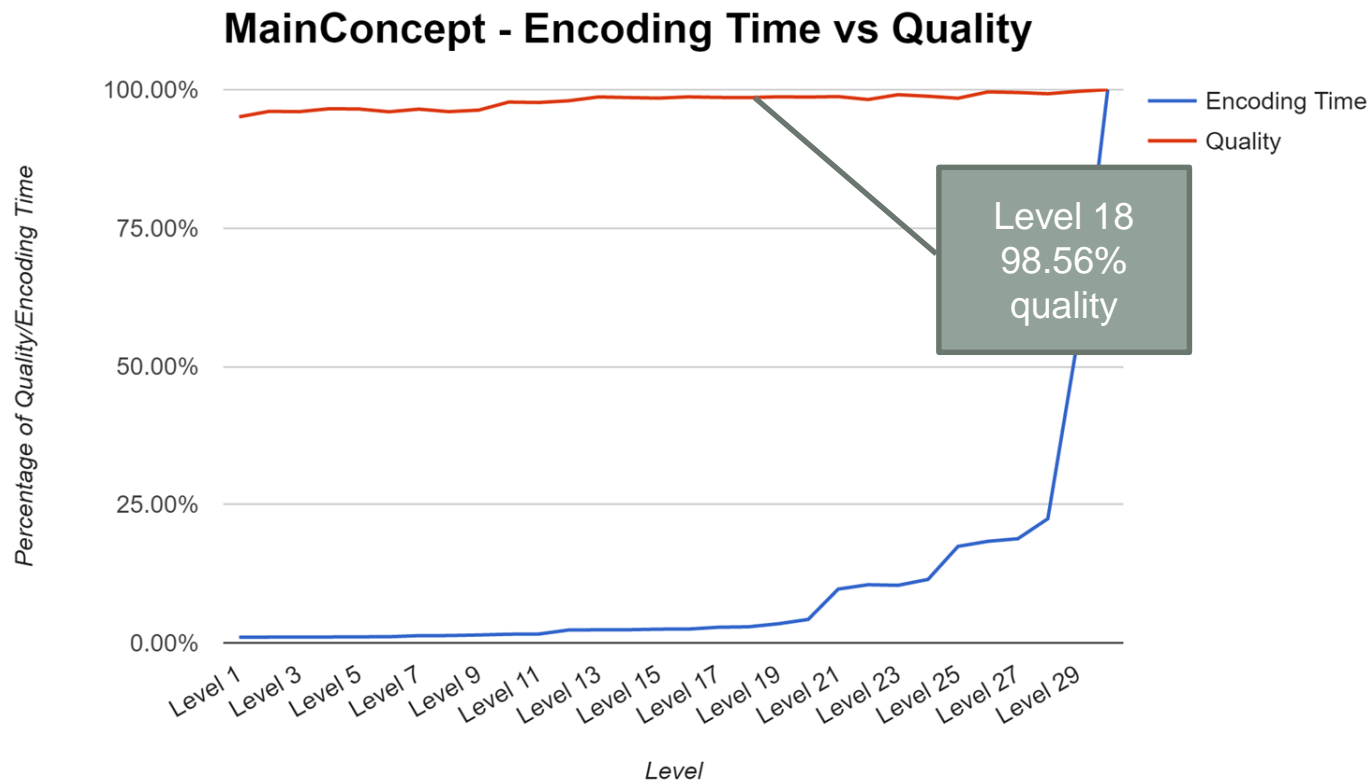
Presets, Quality and Encoding Time

Quality and Encoding Time by HEVC Preset



- Medium is reasonable for quality and throughput
- Superfast for good quality, fast throughput
- Slow for very good quality, reasonable throughput

MainConcept – Times and Quality



Section II: Playback Performance

- How HEVC compares to H.264

Mobil Devices

iOS 11 is compatible with these devices.

iPhone



iPhone X
iPhone 8
iPhone 8 Plus
iPhone 7
iPhone 7 Plus
iPhone 6s
iPhone 6s Plus
iPhone 6
iPhone 6 Plus
iPhone SE
iPhone 5s

iPad



12.9-inch iPad Pro
2nd generation
12.9-inch iPad Pro
1st generation
10.5-inch iPad Pro
9.7-inch iPad Pro
iPad Air 2
iPad Air
iPad
5th generation
iPad mini 4
iPad mini 3
iPad mini 2

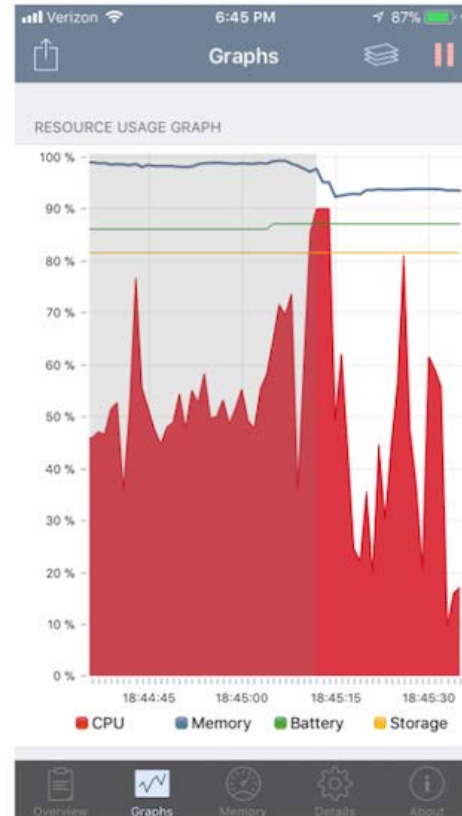
iPod



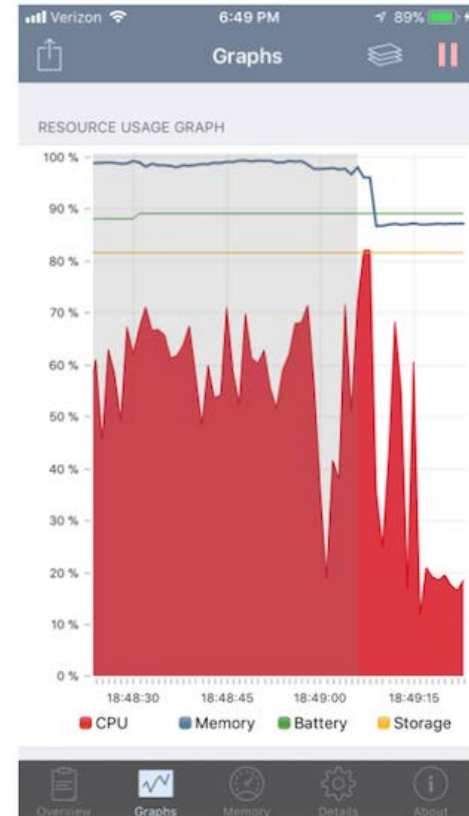
iPod touch
6th generation

Initial Tests – iPhone 6

- H.264 ~ 50%
- HEVC ~ 60%
- Slightly longer battery life with H.264



H.264



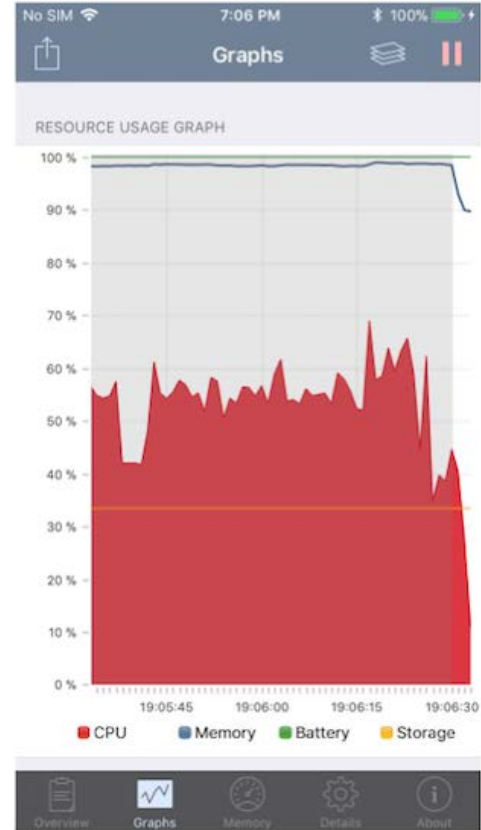
H.265

Initial Tests – iPhone 7

- About the same CPU required during playback
- on iPhone 7+ (and iPads using the same SoC or later, should be very little differential)



H.264



H.265

Desktop Devices

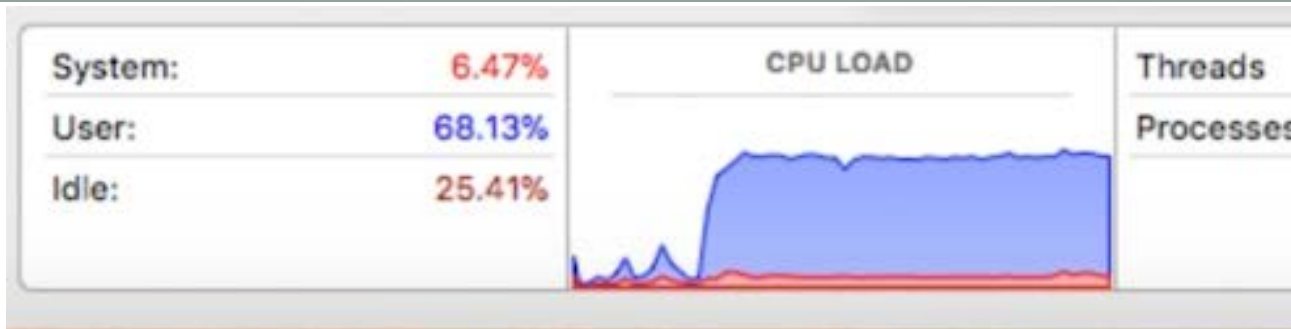
Mac Hardware Requirements

For details about your Mac model, click the Apple icon at the top left of your screen, choose About This Mac. These Mac models are compatible with macOS High Sierra:

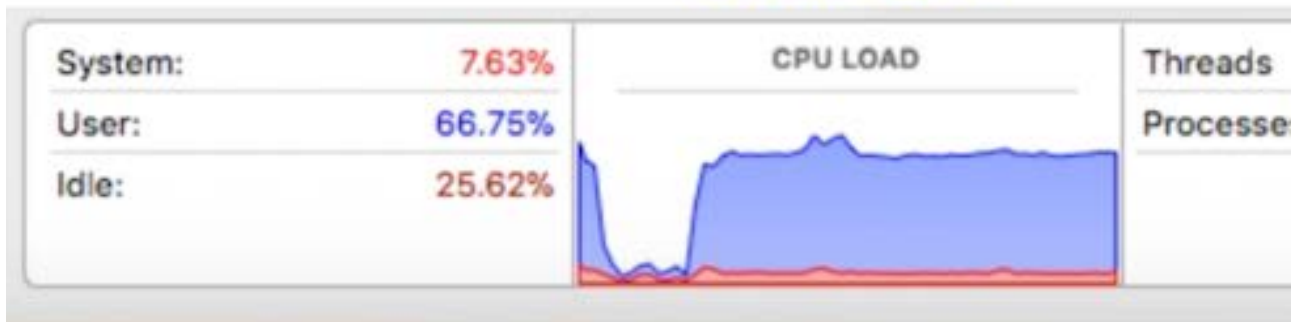
- MacBook (Late 2009 or newer)
- MacBook Pro (Mid 2010 or newer)
- MacBook Air (Late 2010 or newer)
- Mac mini (Mid 2010 or newer)
- iMac (Late 2009 or newer)
- Mac Pro (Mid 2010 or newer)

Initial Tests – 2010 iMac

- About the same (but powerful graphics card)



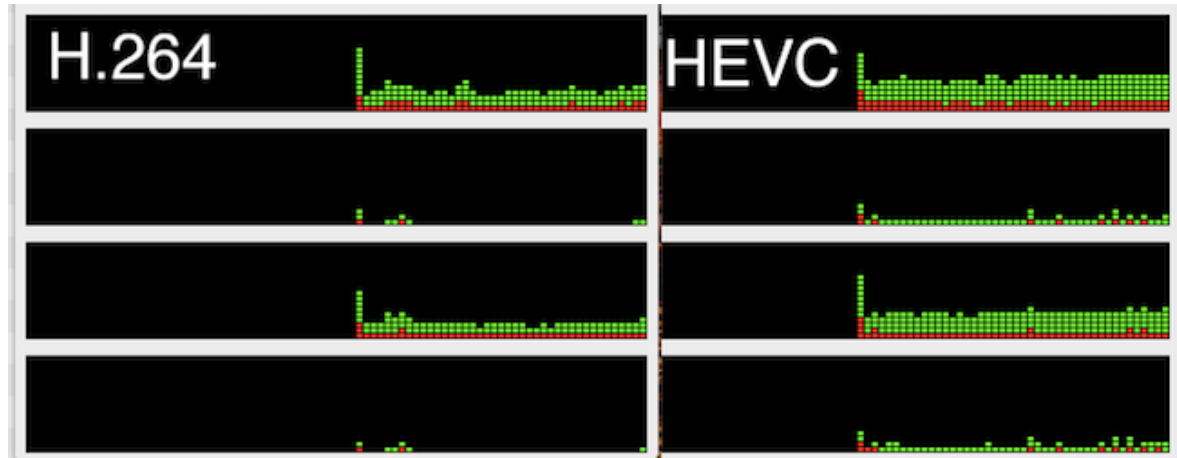
H.264



H.265

Initial Tests – 2011 MacBook Air

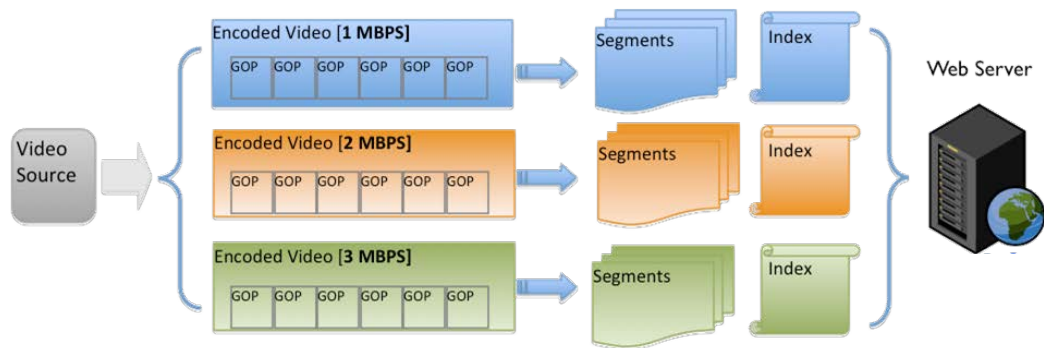
- HEVC about % higher
- Plenty of CPU for HEVC playback



Section III: A Brief Introduction to HLS

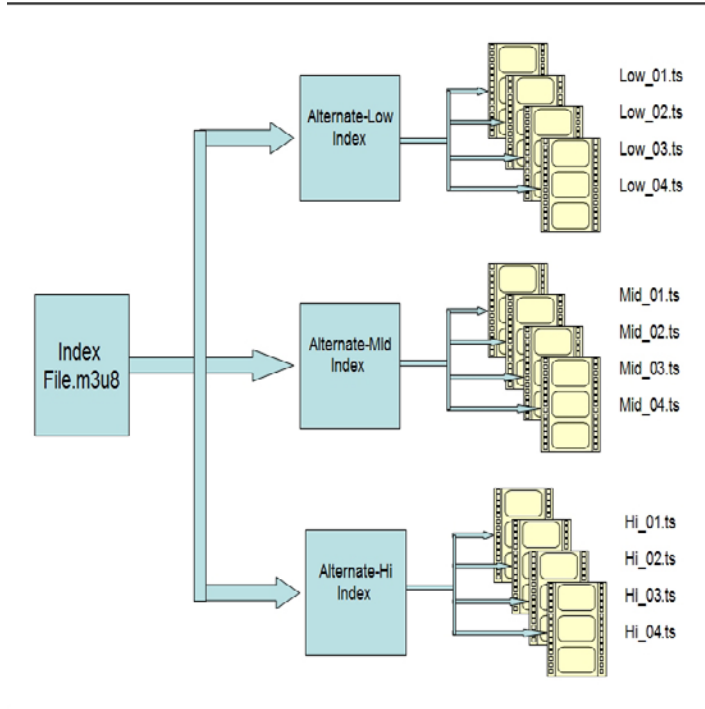
- How HLS works – encoder side
- How HLS works – player side
- HLS content
- HLS manifest files

How HLS Works – Encode Side



- Encoder creates:
 - Multiple sets of segmented video files
 - Index files (M3U8) with file descriptions (res/data rate/profile) and chunk URLs
- Uploads to HTTP web server

How HLS Works - Player Side



- Retrieves master index, retrieves segment from first variant listed in master index
- Monitors the buffer status
- Changes streams as needed using index files to find location
 - If heuristics are good, moves to higher quality stream
 - If heuristics are poor, moves to lower quality stream

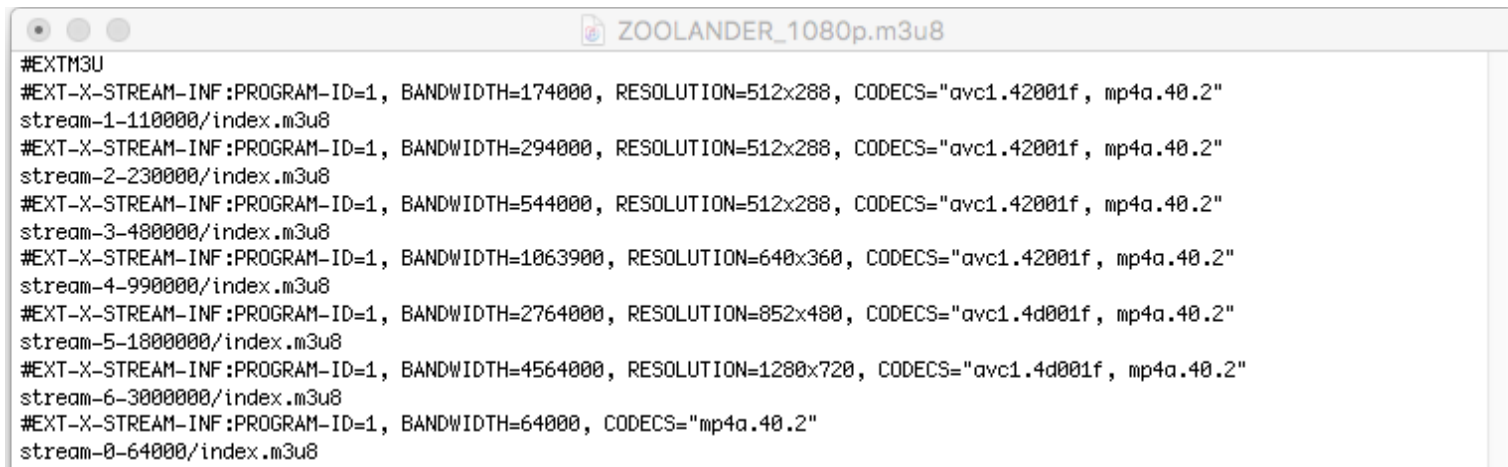
HLS Content

- Initially, used the MPEG-2 transport stream (.ts files)
 - Started with separate files (many, many .ts files)
 - Later enabled byte range requests (more later), enabling player to retrieve segments from a single file
 - Much easier to administrate
- Later, adopted fragmented mp4 files (fMP4)
- HEVC must use fMP4

Manifest or Playlist Files

- **Master**
 - Points to other playlists
- **Variant**
 - One for each piece of content (audio, video, subtitle, caption) in the HLS presentation
 - Points to actual location of content on the server
- **I-frame**
 - Enables trick play, or fast scrubbing backwards and forwards through the file

Master Manifest Files

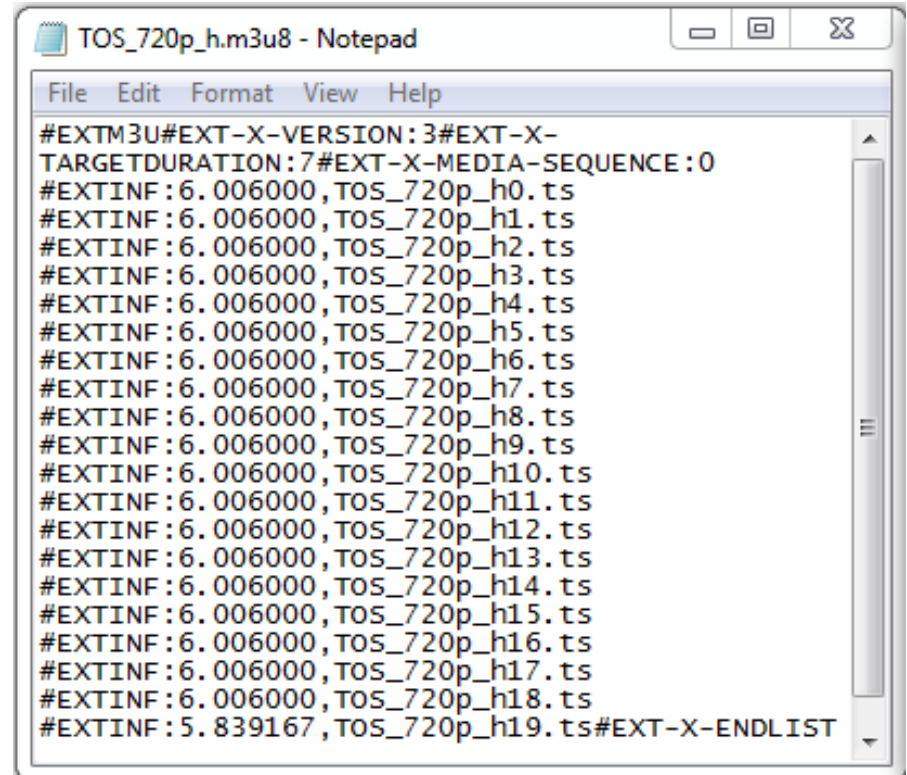


```
#EXTM3U
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=174000, RESOLUTION=512x288, CODECS="avc1.42001f, mp4a.40.2"
stream-1-110000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=294000, RESOLUTION=512x288, CODECS="avc1.42001f, mp4a.40.2"
stream-2-230000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=544000, RESOLUTION=512x288, CODECS="avc1.42001f, mp4a.40.2"
stream-3-480000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=1063900, RESOLUTION=640x360, CODECS="avc1.42001f, mp4a.40.2"
stream-4-990000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=2764000, RESOLUTION=852x480, CODECS="avc1.4d001f, mp4a.40.2"
stream-5-1800000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=4564000, RESOLUTION=1280x720, CODECS="avc1.4d001f, mp4a.40.2"
stream-6-3000000/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=64000, CODECS="mp4a.40.2"
stream-0-64000/index.m3u8
```

- This is the file you link to on your website – first file retrieved
- Contains links to "variant" playlists that identify location of media files
 - Contains enough data to allow player to choose correct streams
 - Codec/profile, resolution, bandwidth

Traditional Variant Playlist

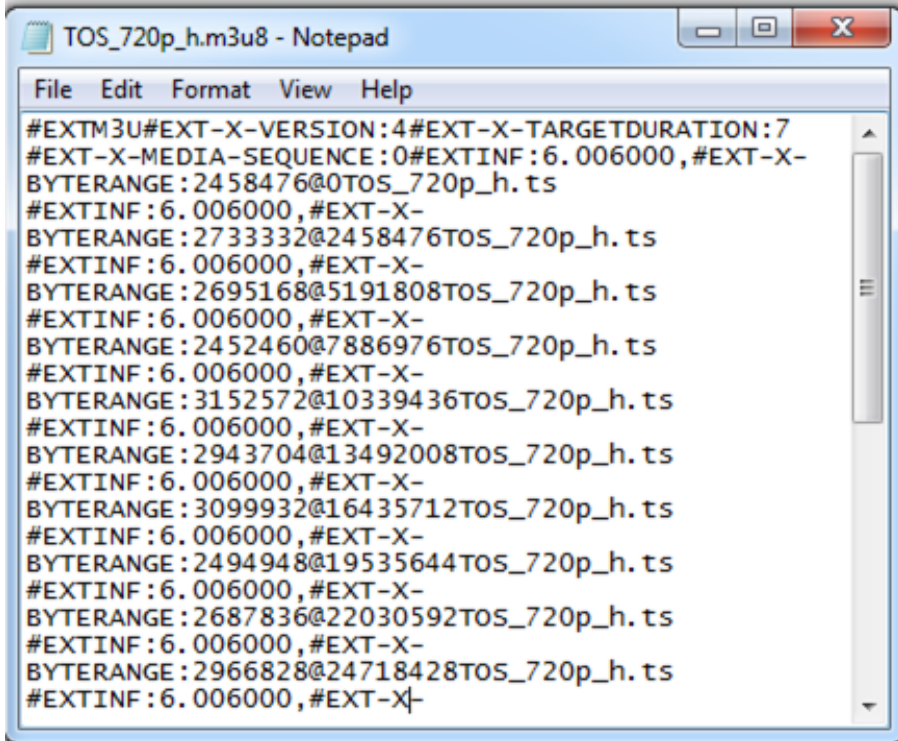
- Name, location, and duration of all individual files
 - .ts files are MPEG-2 transport streams



```
TOS_720p_h.m3u8 - Notepad
File Edit Format View Help
#EXTM3U#EXT-X-VERSION:3#EXT-X-
TARGETDURATION:7#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:6.006000,TOS_720p_h0.ts
#EXTINF:6.006000,TOS_720p_h1.ts
#EXTINF:6.006000,TOS_720p_h2.ts
#EXTINF:6.006000,TOS_720p_h3.ts
#EXTINF:6.006000,TOS_720p_h4.ts
#EXTINF:6.006000,TOS_720p_h5.ts
#EXTINF:6.006000,TOS_720p_h6.ts
#EXTINF:6.006000,TOS_720p_h7.ts
#EXTINF:6.006000,TOS_720p_h8.ts
#EXTINF:6.006000,TOS_720p_h9.ts
#EXTINF:6.006000,TOS_720p_h10.ts
#EXTINF:6.006000,TOS_720p_h11.ts
#EXTINF:6.006000,TOS_720p_h12.ts
#EXTINF:6.006000,TOS_720p_h13.ts
#EXTINF:6.006000,TOS_720p_h14.ts
#EXTINF:6.006000,TOS_720p_h15.ts
#EXTINF:6.006000,TOS_720p_h16.ts
#EXTINF:6.006000,TOS_720p_h17.ts
#EXTINF:6.006000,TOS_720p_h18.ts
#EXTINF:5.839167,TOS_720p_h19.ts#EXT-X-ENDLIST
```

Variant Manifest - Byte Range Request

- Single content file
 - Easier to administrate
- Playlist points to byte ranges within the file
- Need HLS version 5 compatible player



```
TOS_720p_h.m3u8 - Notepad
File Edit Format View Help
#EXTM3U#EXT-X-VERSION:4#EXT-X-TARGETDURATION:7
#EXT-X-MEDIA-SEQUENCE:0#EXTINF:6.006000,#EXT-X-
BYTERANGE:2458476@TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2733332@2458476TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2695168@5191808TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2452460@7886976TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:3152572@10339436TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2943704@13492008TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:3099932@16435712TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2494948@19535644TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2687836@22030592TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
BYTERANGE:2966828@24718428TOS_720p_h.ts
#EXTINF:6.006000,#EXT-X-
```

I-Frame Manifest

- Separate .m3u8 file
- Can point to existing media files, or be a video file with I-frames
 - Either way, the player scans the I-frame at the start of each segment
- Requires HLS version 5 player

```
#EXTM3U
#EXT-X-VERSION:4
#EXT-X-I-FRAMES-ONLY
...
#EXTINF:4.12,
segment1.ts
#EXTINF:3.56,
#EXT-X-BYTERANGE:7144@47000
segment1.ts
#EXTINF:3.82,
#EXT-X-BYTERANGE:10340@1880
segment2.ts
```


Section IV: Specification Overview

- Controlling and sample documents
- Producing HLS streams
 - H264 only
 - H264/HEVC
 - H264/HEVC/HDR

Apple Resources

- HLS Authoring Spec provides
 - Sample encoding ladders
 - Details regarding all aspects of HLS production
- HTTP Live Streaming Examples
 - Provides sample streams and manifest files
- We will reference both during presentation

HLS Authoring Specification for Apple Devices

About HLS Authoring

About HLS Authoring

About HLS Authoring

http://bit.ly/hls_spec_2017

HTTP Live Streaming Examples

http://bit.ly/hls_samps

H.264 Only

Video Streams

- H.264 streams

Trick Play Streams

- i-Frame streams (I-frame playlists (EXT-X-I-FRAME-STREAM-INF) **MUST** be provided to support scrubbing and scanning UI
- SHOULD create one fps “dense” dedicated I-frame renditions
- MAY use I-frames from normal content, but trick play performance is improved with a higher density of I-frames

Configuration (h.264)

- Profile and Level **MUST** be less than or equal to High Profile, Level 4.2.
- SHOULD use High Profile in preference to Main or Baseline Profile

H264 Encoding Ladder - Content

Data Rate	Rez	Frame rate	Profile	Level *	Key Frame	Segment
145	416 x 234	≤ 30 fps	High	4.2	2	6
365	640 x 360	≤ 30 fps	High	4.2	2	6
730	768 x 432	≤ 30 fps	High	4.2	2	6
1100	768 x 432	≤ 30 fps	High	4.2	2	6
2000	960 x 540	Source	High	4.2	2	6
3000	1280 x 720	Source	High	4.2	2	6
4500	1280 x 720	Source	High	4.2	2	6
6000	1920 x 1080	Source	High	4.2	2	6
7800	1920 x 1080	Source	High	4.2	2	6

* Level: Should not use a higher level than required for content resolution and frame rate

H264 Encoding Ladder – I-Frame/Trick Play

Data Rate	Rez	Frame rate	Profile	Key Frame	Profile	Segment
45	640 x 360	1 fps	High	1	High	1
90	768x432	1 fps	High	1	High	1
250	960 x 540	1 fps	High	1	High	1
375	1280 x 720	1 fps	High	1	High	1
580	1920 x 1080	1 fps	High	1	High	1

HEVC/H.264

Video Streams

- H.265
- H.264 streams (For backward compatibility some video content **SHOULD** be encoded with H.264)

Trick Play Streams

- H.264
- H.265 (not specified, but Apple has for both)
- Dedicated encodes are preferred, but can use existing file

Configuration (HEVC)

- Main 10, Level 5, High
 - Level 5 peaks at 30 fps (says same as source)
 - Apple TV won't play
 - Apple HLS sample stream @ 60 fps (but peak at 1080p)
 - Encoding ladder says 30 fps
- Must be fragmented MP4

HEVC Encoding Ladder - Content

Data Rate	Rez	Frame rate	Profile	Level *	Key Frame	Segment
145	640 x 360	≤ 30 fps	Main 10	5.0	2	6
300	768 x 432	≤ 30 fps	Main 10	5.0	2	6
600	960 x 540	≤ 30 fps	Main 10	5.0	2	6
900	960 x 540	≤ 30 fps	Main 10	5.0	2	6
1600	960 x 540	Source	Main 10	5.0	2	6
2400	1280 x 720	Source	Main 10	5.0	2	6
3200	1280 x 720	Source	Main 10	5.0	2	6
4500	1920 x 1080	Source	Main 10	5.0	2	6
5800	1920 x 1080	Source	Main 10	5.0	2	6
8100	2566x1440	Source	Main 10	5.0	2	6
11600	3840x2160	Source	Main 10	5.0	2	6
16800	3840x2160	Source	Main 10	5.0	2	6

* Level: Should not use a higher level than required for content resolution and frame rate

HEVC Encoding Ladder – I-Frame/Trick Play

Data Rate	Rez	Frame rate	Profile	Key Frame	Profile	Segment
40	768 x 432	1 fps	High	1	High	1
75	960 x 540	1 fps	High	1	High	1
200	960 x 540	1 fps	High	1	High	1
300	1280 x 720	1 fps	High	1	High	1
525	1920 x 1080	1 fps	High	1	High	1

Note: 6.1 – I-frame playlists MUST be provided to support scrubbing and scanning UI.
No requirement for HEVC

HDR/HEVC/H264

Video Streams

- HDR
- H.265 (SDR streams must be provided – not specified if H.264 content suffices)
- H.264 streams (For backward compatibility some video content **SHOULD** be encoded with H.264)

Trick Play Streams

- H.264
- H.265 (SDR **must** be provided; not clear if H.264 suffices)
- If HDR provided, should provide at all resolutions

Configuration (HDR)

- MUST be HDR10 or Dolby Vision
 - Dolby Vision – profile 5 (10-bit single layer), level 7
- If HDR provided, **SHOULD** be provided at all resolutions
- 30 fps or less
- Must be fMP4

HDR Encoding Ladder - Content

Data Rate	Rez	Frame rate	Profile	Level *	Key Frame	Segment
160	640 x 360	≤ 30 fps	Main 10	5.0	2	6
336	768 x 432	≤ 30 fps	Main 10	5.0	2	6
730	960 x 540	≤ 30 fps	Main 10	5.0	2	6
1090	960 x 540	≤ 30 fps	Main 10	5.0	2	6
1930	960 x 540	Source	Main 10	5.0	2	6
2900	1280 x 720	Source	Main 10	5.0	2	6
3850	1280 x 720	Source	Main 10	5.0	2	6
5400	1920 x 1080	Source	Main 10	5.0	2	6
7000	1920 x 1080	Source	Main 10	5.0	2	6
9700	2566x1440	Source	Main 10	5.0	2	6
13900	3840x2160	Source	Main 10	5.0	2	6
20000	3840x2160	Source	Main 10	5.0	2	6

* Level: Should not use a higher level than required for content resolution and frame rate

HDR Encoding Ladder – I-Frame/Trick Play

Data Rate	Rez	Frame rate	Profile	Key Frame	Profile	Segment
55	768 x 432	1 fps	High	1	High	1
94	960 x 540	1 fps	High	1	High	1
238	960 x 540	1 fps	High	1	High	1
360	1280 x 720	1 fps	High	1	High	1
650	1920 x 1080	1 fps	High	1	High	1

Note: 6.1 – I-frame playlists MUST be provided to support scrubbing and scanning UI.
No requirement for HEVC

All Frame Rate/Bitrate Control

- Frame rates above 60 fps **SHALL NOT** be used.

VOD:

- If progressive use that rate
- You **SHOULD** de-interlace 30i content to 60p instead of 30p (streams above 2 Mbps)

Live:

- Live/linear video from NSTC or ATSC source **SHOULD** be 60 or 59.94 fps (PAL=50 fps)
- HEVC/HDR – max 30 fps

VOD:

- Average segment bit rate **MUST** be within 10% of the AVERAGE-BANDWIDTH attribute
- Measured peak bit rate **MUST** be within 10% of the BANDWIDTH attribute.
- Peak bit rate **SHOULD** be no more than 200% of the average bit rate.

Live:

- Average segment bit rate over a long (~1 hour) **MUST** be less than 110% of the AVERAGE-BANDWIDTH attribute
- Measured peak bit rate **MUST** be less than 125% of the BANDWIDTH attribute.

Apple's HEVC/H264 Encoding Ladder

- Nine HEVC video variants
 - Gear 9 - 1920x1080 @ 5.8 Mbps
 - Gear 8 - 1920x1080 @ 4.5 Mbps
 - Gear 7 - 1920x1080 @ 3.2 Mbps
 - Gear 6 - 1280x720 @ 2.4 Mbps
 - Gear 5 - 960x540 @ 1.7 Mbps
 - Gear 4 - 768x432 @ 990 Mbps
 - Gear 3 - 640x360 @ 660 kbps
 - Gear 2 - 480x270 @ 350 kbps
 - Gear 1 - 416x234 @ 145 kbps
- I-frame variants in HEVC/H264 formats
- Nine H.264 video variants
 - Gear 9 - 1920x1080 @ 7.8 Mbps
 - Gear 8 - 1920x1080 @ 6.0 Mbps
 - Gear 7 - 1920x1080 @ 4.5 Mbps
 - Gear 6 - 1280x720 @ 3.0 Mbps
 - Gear 5 - 960x540 @ 2.0 Mbps
 - Gear 4 - 768x432 @ 1.1 Mbps
 - Gear 3 - 640x360 @ 730 kbps
 - Gear 2 - 480x270 @ 365 kbps
 - Gear 1 - 416x234 @ 145 kbps
- Dolby obviously not required
- I-Frame variants (fast-forward / rewind support)
- 3 audio renditions
 - AAC-LC - 48 kHz stereo @ 160 kbps
 - AC-3 - 48 kHz 5.1 @ 384 kbps
 - EC-3 - 48 kHz 5.1 @ 192 kbps
- 1 subtitle rendition (WebVTT)
 - English

H.264 Adaptive Group (from Master)

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=2190673,BANDWIDTH=2523597,CODECS="avc1.640020,mp4a.40.2",
RESOLUTION=960x540,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v5/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=8052613,BANDWIDTH=9873268,CODECS="avc1.64002a,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v9/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=6133114,BANDWIDTH=7318337,CODECS="avc1.64002a,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v8/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=4681537,BANDWIDTH=5421720,CODECS="avc1.64002a,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v7/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=3183969,BANDWIDTH=3611257,CODECS="avc1.640020,mp4a.40.2",
RESOLUTION=1280x720,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v6/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=1277747,BANDWIDTH=1475903,CODECS="avc1.64001f,mp4a.40.2",
RESOLUTION=768x432,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v4/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=890848,BANDWIDTH=1017705,CODECS="avc1.64001f,mp4a.40.2",
RESOLUTION=640x360,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v3/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=533420,BANDWIDTH=582820,CODECS="avc1.64001f,mp4a.40.2",
RESOLUTION=480x270,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v2/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=303898,BANDWIDTH=339404,CODECS="avc1.64001f,mp4a.40.2",
RESOLUTION=416x234,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v1/prog_index.m3u8

H.264 I-Frame Group

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=928091,BANDWIDTH=1015727,CODECS="avc1.640028",
RESOLUTION=1920x1080,URI="tp5/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=731514,BANDWIDTH=760174,CODECS="avc1.64001f",
RESOLUTION=1280x720,URI="tp4/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=509153,BANDWIDTH=520162,CODECS="avc1.64001f",
RESOLUTION=960x540,URI="tp3/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=176942,BANDWIDTH=186651,CODECS="avc1.64001f",
RESOLUTION=640x360,URI="tp2/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=90796,BANDWIDTH=95410,CODECS="avc1.64001f",
RESOLUTION=480x270,URI="tp1/iframe_index.m3u8"

H.265 Adaptive Group (from Master)

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=1966314,BANDWIDTH=2164328,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=960x540,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v14/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=6105163,BANDWIDTH=6664228,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v18/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=4801073,BANDWIDTH=5427899,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v17/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=3441312,BANDWIDTH=4079770,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=1920x1080,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v16/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=2635933,BANDWIDTH=2764701,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=1280x720,FRAME-RATE=60.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v15/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=1138612,BANDWIDTH=1226255,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=768x432,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v13/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=829339,BANDWIDTH=901770,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=640x360,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v12/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=522229,BANDWIDTH=548927,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=480x270,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v11/prog_index.m3u8

#EXT-X-STREAM-INF:AVERAGE-BANDWIDTH=314941,BANDWIDTH=340713,CODECS="hvc1.2.4.L123.B0,mp4a.40.2",
RESOLUTION=416x234,FRAME-RATE=30.000,CLOSED-CAPTIONS="cc",AUDIO="a1",SUBTITLES="sub1"v10/prog_index.m3u8

HEVC I-Frame Group

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=287207,BANDWIDTH=328352,CODECS="hvc1.2.4.L123.B0",
RESOLUTION=1920x1080,URI="tp10/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=216605,BANDWIDTH=226274,CODECS="hvc1.2.4.L123.B0",
RESOLUTION=1280x720,URI="tp9/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=154000,BANDWIDTH=159037,CODECS="hvc1.2.4.L123.B0",
RESOLUTION=960x540,URI="tp8/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=90882,BANDWIDTH=92800,CODECS="hvc1.2.4.L123.B0",
RESOLUTION=640x360,URI="tp7/iframe_index.m3u8"

#EXT-X-I-FRAME-STREAM-INF:AVERAGE-BANDWIDTH=50569,BANDWIDTH=51760,CODECS="hvc1.2.4.L123.B0",
RESOLUTION=480x270,URI="tp6/iframe_index.m3u8"

Our Playback Experiment

- Hybrid and low hybrid
- Tests and results
- Conclusions

Hybrid – Both Complete Ladders

16:9 aspect ratio	H.264/AVC
416 x 234	145
640 x 360	365
768 x 432	730
768 x 432	1100
960 x 540	2000
1280 x 720	3000
1280 x 720	4500
1920 x 1080	6000
1920 x 1080	7800

16:9 aspect ratio	HEVC/H.265 30 fps
640 x 360	145
768 x 432	300
960 x 540	600
960 x 540	900
960 x 540	1600
1280 x 720	2400
1280 x 720	3400
1920 x 1080	4500
1920 x 1080	5800
2560 x 1440	8100
3840 x 2160	11600
3840 x 2160	16800

Low Hybrid

16:9 aspect ratio	H.264/AVC
416 x 234	145
640 x 360	365
768 x 432	730
768 x 432	1100
960 x 540	2000
1280 x 720	3000
1280 x 720	4500
1920 x 1080	6000
1920 x 1080	7800

16:9 aspect ratio	HEVC/H.265 30 fps
640 x 360	145
768 x 432	300
960 x 540	600
960 x 540	900
960 x 540	1600
1280 x 720	2400
1280 x 720	3400
1920 x 1080	4500
1920 x 1080	5800
2560 x 1440	8100
3840 x 2160	11600
3840 x 2160	16800

Results

- 43 desktop
- 19 mobile

What Did We Learn

- Excellent performance and compatibility
 - H.264 streams played on older devices without problem
 - Very few quality issues
 - No disruption when switching between H.264 and HEVC
-

What Did We Learn

- Playback

- Apple typically won't retrieve higher resolution file than display resolution
 - One instance where MBP with 1800 vertical rez retrieved 4K file
- 4K doesn't get retrieved all that often
 - Average bandwidth was 580 Mbps
 - Lowest was 64 kbps for 16.8 Mbps stream
 - Many devices with very high bandwidth and necessary resolution could not play

Ladder Composition

- “May” make a difference
- There were several instances where the result between hybrid and low hybrid differed
 - A different max file was retrieved
 - In all but one instance, the experience was worse
 - Either H.264 instead of HEVC
 - Lower data rate
 - More analysis necessary

What Know About Switching?

- Ask Apple – two streams in ladder; which does player select?

1080p – HEVC – 2 Mbps

720p – H.264 – 2.5 Mbps

- Logic in transition but “knows” H.265 should be higher quality than H.264 at similar data rates
 - So don’t need to game the system (create artificially high data rate for H.265 streams so
- Typically won’t switch between H.264 and H.265 when both available
- Apple recommends full H.264/H.265 ladders in all cases

Section V. Producing HEVC/HLS

- DIY – VOD
 - FFmpeg – create the A/V files
 - Bento4 – package and manifest files
- Third party alternatives
 - Live
 - VOD

Creating HEVC Files in FFmpeg

- Use the x265 codec
 - Widely recognized as one of the fastest and highest quality
 - Need to compile Main10-specific version
- All scaling and other syntaxes apply
- Need to choose profile and preset (unless defaults OK)
- Must use `-x265-params` command for some parameters

Encoding x265 in FFmpeg

```
ffmpeg -y -i TOS_1080p.mov -c:v libx265 -preset slow -x265-params profile=main:keyint=48:
min-keyint=48:scenecut=0:ref=5:bframes=3:b-adapt=2:bitrate=4000:vbv-maxrate=4400:vbv-bufsize=4000
-an -pass 1 -f mp4 NUL && \
```

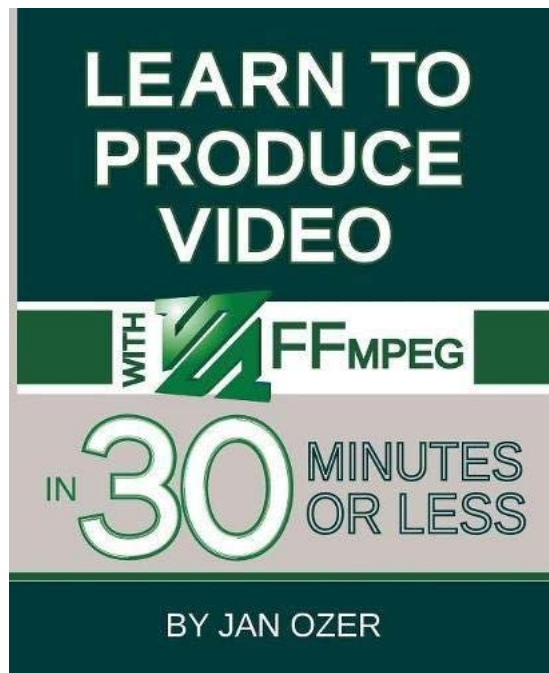
```
ffmpeg -i TOS_1080p.mov -c:v libx265 -preset slow -x265-params profile=main:keyint=48:
min-keyint=48:scenecut=0:ref=5:bframes=3:b-adapt=2:bitrate=4000:vbv-maxrate=4400:vbv-bufsize=4000
-an -pass 2 TOS_1080p_h.mp4
```

```
ffmpeg -i TOS_1080p.mov -c:v libx265 -s 1280x720 -preset slow -x265-params profile=main:
keyint=48: min-keyint=48:scenecut=0:ref=5:bframes=3:b-adapt=2:bitrate=1000:vbv-maxrate=1100:
vbv-bufsize=1000 -an -pass 2 TOS_720p_1.mp4
```

- Integrate x265 commands into FFmpeg
 - x265-params – start of x265 commands, in x265 syntax
 - <http://x265.readthedocs.io/en/default/>
 - One string of commands, separated by colon, no spaces until finished
- Preset, an (audio no), pass, format, and Null outside of this structure
- Scaling commands outside of –x265-params structure

FFmpeg Learning Resources

- Includes H.264/H.265
 - Creation of variant playlists with FFmpeg
 - Variant/master playlists with Apple tools
 - No Bento
 - No cloud stuff
- D103 - HOW TO: Building a More Robust Cloud Encoder With FFMPEG & More
 - Thus - 1:45 – 2:30



http://bit.ly/ffmpeg_30

Introduction to Bento4

- What it is: A fast, modern, open source C++ toolkit for all your MP4, HLS, and MPEG DASH media format needs
 - <https://www.bento4.com/>
 - Documentation for HLS - <https://www.bento4.com/developers/hls/>
- What you can do with Bento4
- Bento 4 vs. FFmpeg
- HLS options and Bento4 syntax

What can I do with Bento4?

- HLS generation, including master manifests, stream level manifests, mpeg-2 ts files, and fMP4 (fragmented MP4)
- MP4 to fMP4 conversion
- DASH generation
- Parsing and multiplexing of H.264 and AAC streams
- Support for DRM (Marlin, PlayReady, Widevine and FairPlay).
- Support for H.264, H.265, AAC, AC3, eAC3, DTS, ALAC, and other codec types.
- Dual generation of HLS and DASH from fragmented MP4
- Atom/box editing, and stream/codecs information
- A lot more... <https://www.bento4.com/>

Bento4 vs FFmpeg

- Bento4 focuses on MP4 based content: Packaging & Transmuxing
- FFMPEG is a broad spectrum tool for media conversion, encoding & packaging

MP4HLS options

- Note: MP4HLS can only create .ts streams; not fMP4, so you can't use to create HEVC/H264 packages. You'll need to use MP4DASH.
- Options
 - Master playlists
 - Single file output with byte range requests
 - I-Frame only playlists
 - AES encryption
 - DRM
 - Audio stream sidecar
 - Subtitle sidecar
 - fMP4

Create Multiple Bitrate Assets

```
mp4hls --hls-version 4 input_7000kb.mp4 input_5000kb.mp4 input_3500kb.mp4
```

- Outputs:
- Master.m3u8
- Stream.m3u8 for each bitrate
- Iframe.m3u8 for each bitrate
- ts fragments for each bitrate

Multiple Audio Streams

```
mp4hls video.mp4 spanish_audio.m4a (different audio file)
```

```
mp4hls video.mp4 [+language=es]audio.m4a (multiplexed audio file, getting the spanish  
stream)
```

Outputs:

- Master.m3u8
- Stream.m3u8 for video and audio
- Iframe.m3u8 for video and audio
- ts fragments
- Audio.m3u8 and aac fragments

WebVTT Subtitles

```
mp4hls video.mp4 [+format=webvtt,+language=en]english.vtt
```

Outputs

- Master.m3u8
- Stream.m3u8
- Webvtt manifest and .vtt file

Encryption and Single Segment

```
mp4hls --hls-version 4 --output-single-file --segment-duration 6 --encryption-mode AES-128  
--encryption-key abaa09cd8c75abba54ac12dbcc65acd7 --encryption-url  
http://getmyKey?token=token video.mp4
```

Outputs

- All HLS assets (master, stream with byterange requests, iframe, single ts file)
- Assets are encrypted with AES-128, and encryption URL is added to the stream manifests
- Segment duration will be set to 6 seconds, but will only segment at the closest i-frame

Dual HLS and DASH From fMP4

- Three step workflow with MP4DASH

- Encode in FFmpeg
- Convert each MP4 to fMP4 (mp4fragment)
- Create package (mp4dash)

- `mp4fragment input.mp4 output.mp4` (converts mp4 to fmp4)
- `mp4dash --force --hls --no-split --use-segment-timeline output.mp4` (without `--no-split` it will output .m4s segments)

- Outputs

- Master.m3u8
- Audio.m3u8
- Video.m3u8
- Stream.mpd (DASH manifest)

Example Master Playlist for Single Bitrate

```
#EXTM3U
#EXT-X-VERSION:6
# Media Playlists
# Audio
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-
  ID="audio/mp4a",LANGUAGE="en",NAME="English",AUTOSELECT=YES,DEFAULT=YES,URI="audio-en-mp4a.m3u8"
# Video
#EXT-X-STREAM-INF:AUDIO="audio/mp4a",AVERAGE-
  BANDWIDTH=3454711,BANDWIDTH=4209761,CODECS="avc1.640020,mp4a.40.2",RESOLUTION=1280x720 video-
  avc1.m3u8
```

Other Info

- Bento will only segment at an i-frame
- Creates HLS assets faster than ffmpeg or shaka packager
- Gathers its metadata while segmenting, so codecs, average bandwidth, bandwidth, and resolution are automatically added to the manifests
- A full set of DASH and metadata options

List of all Bento4 binaries: <https://www.bento4.com/>

Implementing Steps

- VOD: Server-based HEVC/HLS asset generation
- Cloud workflow
- Scaling
- Cloud encoding (the server)

VOD: Server-based HEVC/HLS Asset Generation

- Overview
- Sizing your server
- Our experience
- Hardware starting point
- GPU pipeline
- Getting the software

OVERVIEW

- Choose your Cloud:
 - AWS
 - Azure
 - RackSpace
 - IBM SoftLayer
- Or don't (On-prem)
- Or a hybrid (e.g. - On-prem and S3)

SIZING YOUR SERVER

- General
 - What general bitrates are you dealing with?
- Live
 - How many concurrent live streams?
 - Are you also transcoding optional renditions for ABR?
- VOD
 - How many concurrent videos being processed?
 - Is it transcoding or just transmuxing?
 - Do you need to create sidecar assets?

OUR EXPERIENCE

- In AWS we've found m3.large to be a pretty cost effective, decently performant and reliable instance size
- We made our decision in Azure based on AWS and went with as similar a match we could find, DS2_V2
- We use Linux as our base since it's friendlier with our software stack. Mostly RHEL.

STARTING POINT

- Get started with ec2 instances:
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html
- Get started with Azure VMs: <https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-linux-quick-create-portal/>

GPU PIPELINE

- Offload processing from CPU to dedicated hardware
- FFmpeg has some support for GPU Acceleration
- You need to have specific supported hardware
 - Example: AWS EC2 g2.2xlarge + CUDA + FFmpeg with -hwaccel option specified

GETTING THE SOFTWARE

- You'll need to download and install software
- Our preferred toolset:
 - Bento4/FFmpeg (Video processing and Static Builds are easy install)
 - ImageMagick (spritesheets, thumbnails and image manipulation)
 - Node.js (You need an application server wrapper)
 - MongoDB (You need some data persistence)
 - Cloud Provider SDK (e.g. AWS SDK for JavaScript in Node.js)

Cloud Workflow: Making it Happen

- Designing a workflow API
- Workflow: file transfer
- Workflow: queue
- Open source libraries
- Sample code

DESIGNING A WORKFLOW - API

- You need a good workflow architecture
- Similar to AWS Simple Workflow Service for logical and atomic chunks:
 - Workflow (End to End Execution)
 - Steps (Ingestion, Processing, Transfer)
 - Tasks (Create alternate bitrate rendition, Thumbnails)
 - Adaptors (We added this to be agnostic.
E.g. AWS S3 vs. Azure Blob vs. On-prem)

WORKFLOW: FILE TRANSFER

- Try to leverage any performance enhancements available
- Day to Day Ingestion
 - AWS Multipart Upload
 - Azure Streaming Put a BlockBlob
- Initial Content Migration
 - AWS Import/Export Snowball
 - Azure Import/Export Service

WORKFLOW: QUEUE

- Gracefully handle all your users
- Processing takes time. You need to line up requests.
- Queuing w/persistence also lets you keep track of job status and what's pending in case of restart.

OPEN SOURCE LIBRARIES

- When there's a vibrant community you never have to reinvent the wheel
- We use Node.js which has node modules.
 - aws-sdk: AWS JavaScript Library for Node.js
 - fluent-ffmpeg: A node wrapper for the FFmpeg command line tool
 - q: A node promise library
 - async: Asynchronous JavaScript helper

SAMPLE CODE

- Check out the demo: <https://github.com/realeyes-media/demo-encoder>
- Here's a snippet

```
input.inputOptions = options.inputOptions;
output.outputOptions = ["-hls_time 8", "-hls_list_size 0", "-bsf:v h264_mp4toannexb", "-threads
0"];
input.inputURI = path.join(__dirname, '../..' + options.inputURI);
output.outputURI = directory + '/' + options.fileName + options.timestamp + '_' + bitrate + '.'
+ options.outputType;
options.outputURI = output.outputURI;
output.outputOptions.push('-b:v ' + bitrate + 'k', '-r ' + options.fps);

// Use options to call ffmpeg executions in parallel
executeFfmpeg(input, output)
```

Scaling

- Scaling and concurrency
- Scaling – multiple instances
- Multi-instance balancing
- Auto-scaling
- Container swarms

SCALING & CONCURRENCY

- How high can we go?
FFmpeg will not error when the CPU is busy, just takes longer to process.
- First - Determine the Scenario:
 - The volume of files you need to simultaneously process
 - The average size of the files you need to process
 - The processing time that's acceptable for you org
 - The kinds of operations that need to occur (e.g. Just transmux? Transcode to 4 renditions?)
- Second - Run Performance Tests

SCALING - MULTIPLE INSTANCES

- Bigger instance or more instances?
- Bigger Instance
 - PRO: Handles more concurrency
 - CONS: Can be more costly
- More Instances
 - PRO: Cheaper - Can be scaled up and down to only pay when needed
 - CONS: More complicated to manage

MULTI INSTANCE BALANCING

- Scale Horizontally Transparently
Clients hit a load balancer
- You can add more instances as needs grow in a transparent and simple way
- If your architecture is sound there's no need for session stickiness between the clients and the transcoding system
- AWS Elastic Load Balancer:
<https://aws.amazon.com/elasticloadbalancing/>
- Azure Load Balancing: <https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-linux-load-balance/>

AUTO-SCALING

- Leverage Auto Scaling Features
- Automate the spin up/down of instances based on a number of criteria:
 - Instance Load
 - Periodic Need for Faster Processing
 - Time of Day
 - Specific Events
- AWS Auto Scaling: <https://aws.amazon.com/autoscaling>
- Azure Auto Scale: <https://azure.microsoft.com/en-us/documentation/articles/cloud-services-how-to-scale-portal/>

CONTAINER SWARMS

- Docker is all the rage. Swarms and Service Discovery
- Create a swarm of Docker containers for a highly repeatable processing server snapshot that utilizes system resources efficiently
- Further increase automation through service discovery
- Implement “auto scaling” on steroids

Cloud Encoding (The Server)

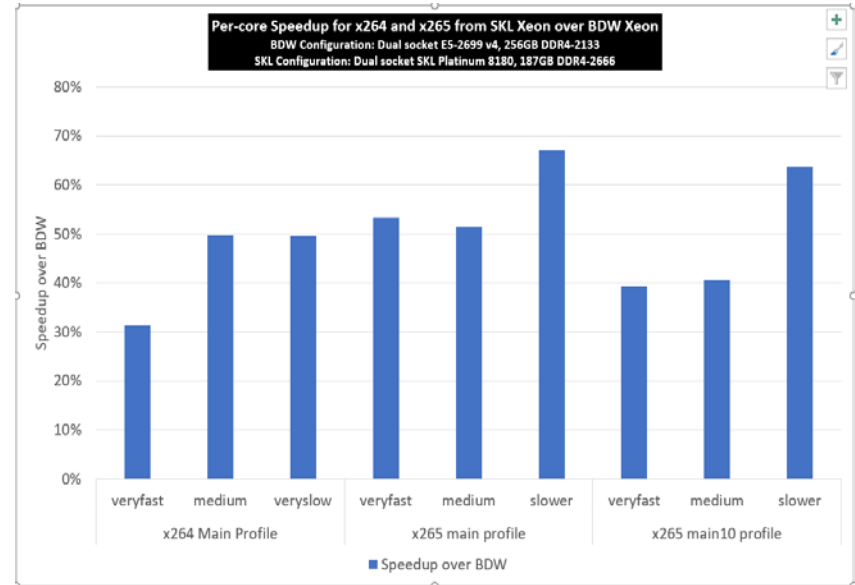
- >>> DEMO <<<

LIVE: Streaming with HEVC/HLS

- x265 Boost from Intel Xeon Scalable processor family
- Encoding – basically it comes down to hardware or cloud

HEVC Live – Intel Scalable Processor Family

- [x265 Boost from Intel Xeon Scalable Processor Family](#)
- x265 show a 67% average per-core gain for encoding using HEVC Main profile
- 50% average gain with Main10 profile across different presets



HEVC Live

- [Live 4K HEVC/H.265 Software Encoding](#)
- Haivision demoed live 4Kp60 HEVC software-only (x265) performance video streaming w/off the shelf hardware
- In the end it all comes down to hardware for live

More Demos

- Manifest Demo
- Playback demo and discussion (H.265 only)
- Playback demo and discussion (mixed H.264 and H.264)
- Playback demo and discussion (H.264 only)
- Additional resources

Manifest Demo: Walking through VOD and LIVE HEVC/HLS during playback (manifest viewer)

Manifest Demo: Walking through VOD and LIVE HEVC/HLS during playback (manifest viewer)

Playback Demo/Discussion: H.265 only

Playback Demo/Discussion: Mixed H.265 + H.264

Playback Demo/Discussion: H.264 only

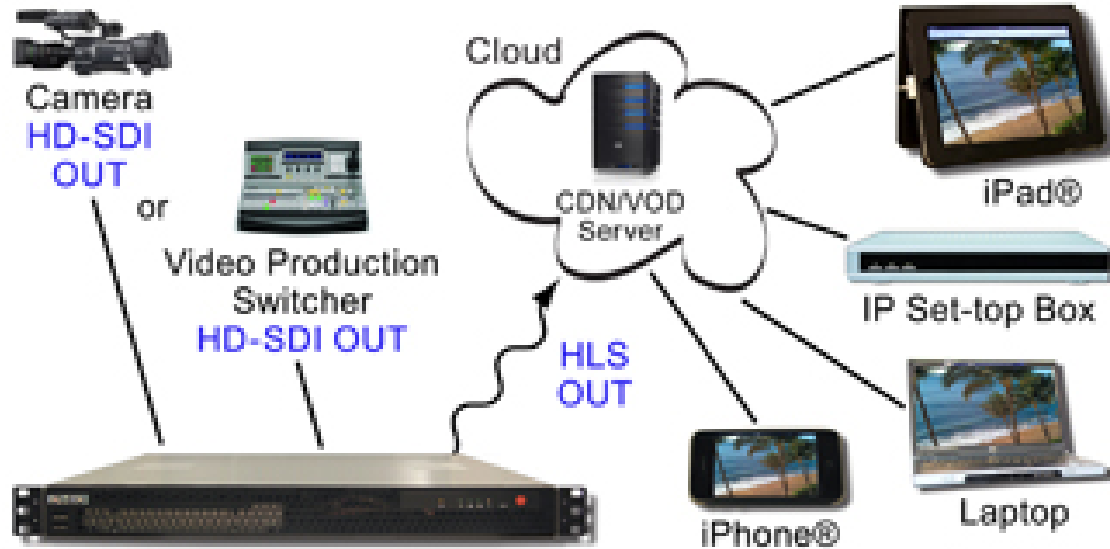
Third Party Alternatives

- **Live**
 - Full transcode and package
 - Contribution
 - Cloud transcode
- **VOD**
 - Appliance
 - Software
 - Cloud

Live: Full Transcode and Package

- Input video, output encoded/packaged files for direct delivery, or sending to CDN
- Products (not an exhaustive list)
 - DVEO MultiStreamer
 - Elemental Live
 - Harmonic Electra XT
 - Harmonic VOS Cloud Software
 - Telestream Vantage Lightspeed

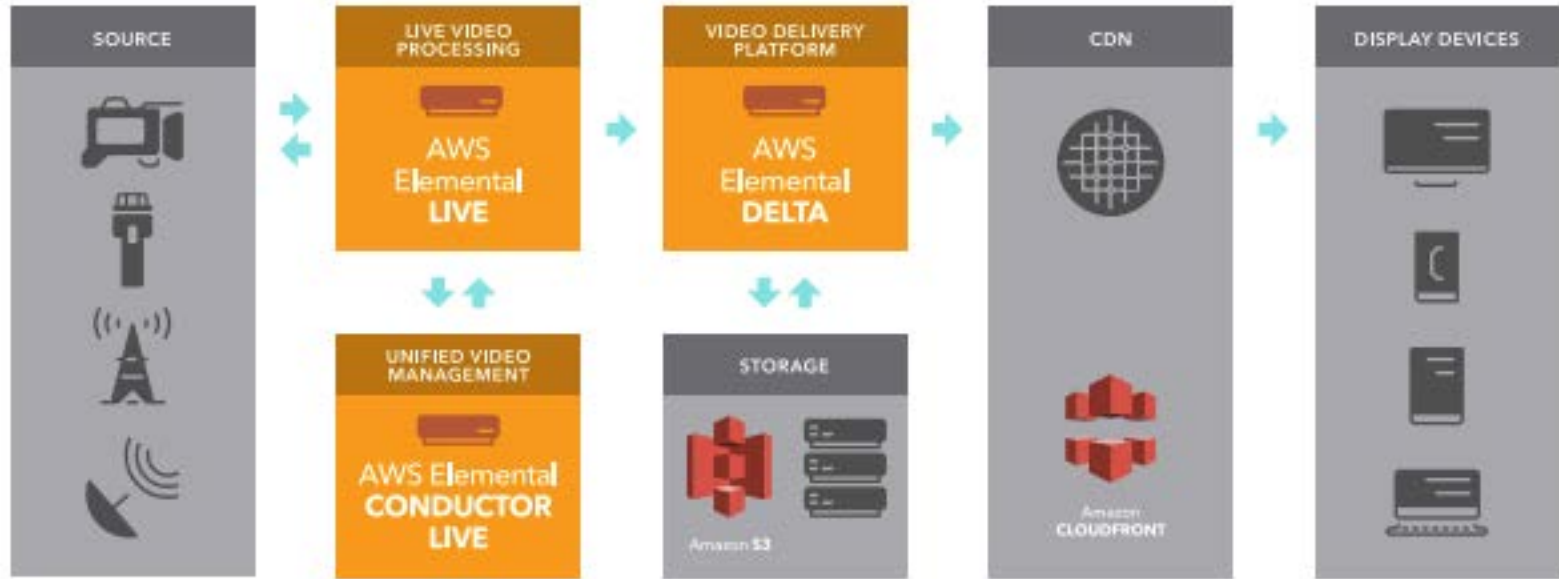
Full Transcode and Package: DVEO MultiStreamer



- Hardware appliance

- No pricing info on website

Full Transcode and Package: Elemental Live



- Software, appliance, or PaaS

- No pricing info on website

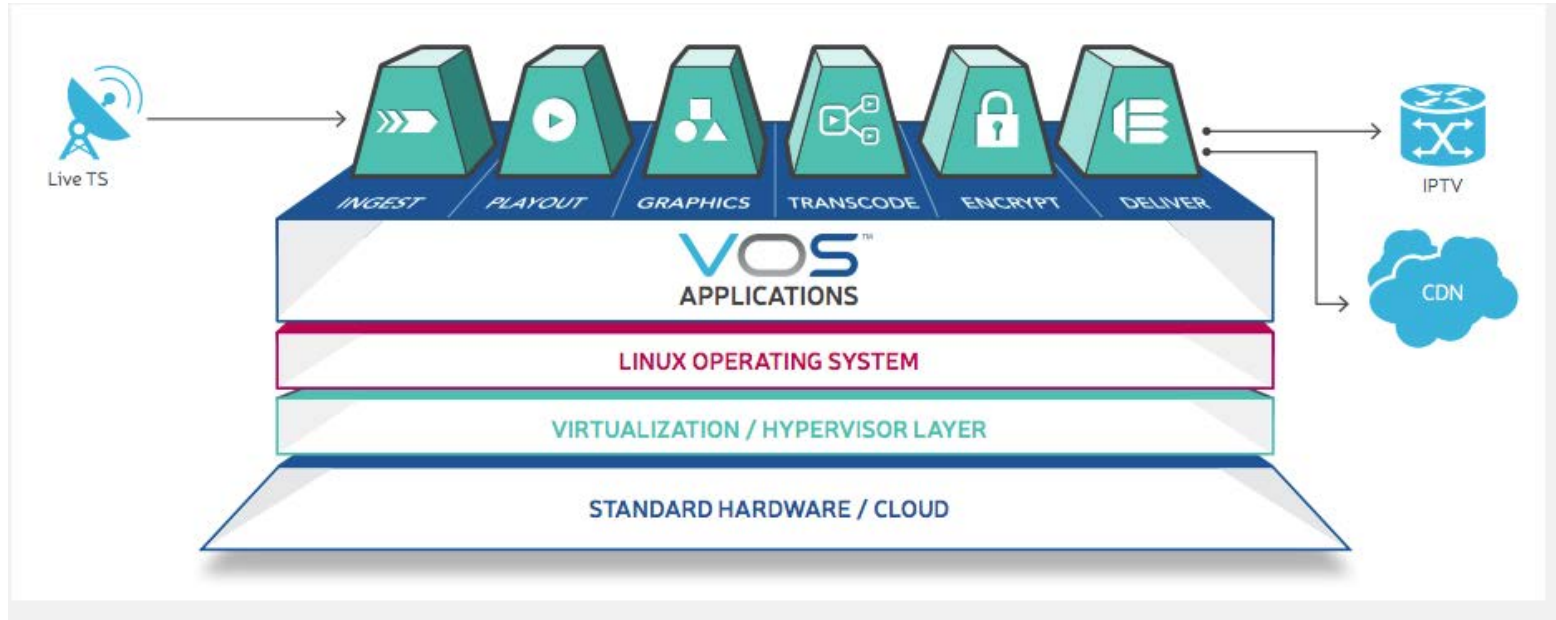
Full Transcode and Package: Harmonic Electra XT, X2, X2S, VS



- Hardware

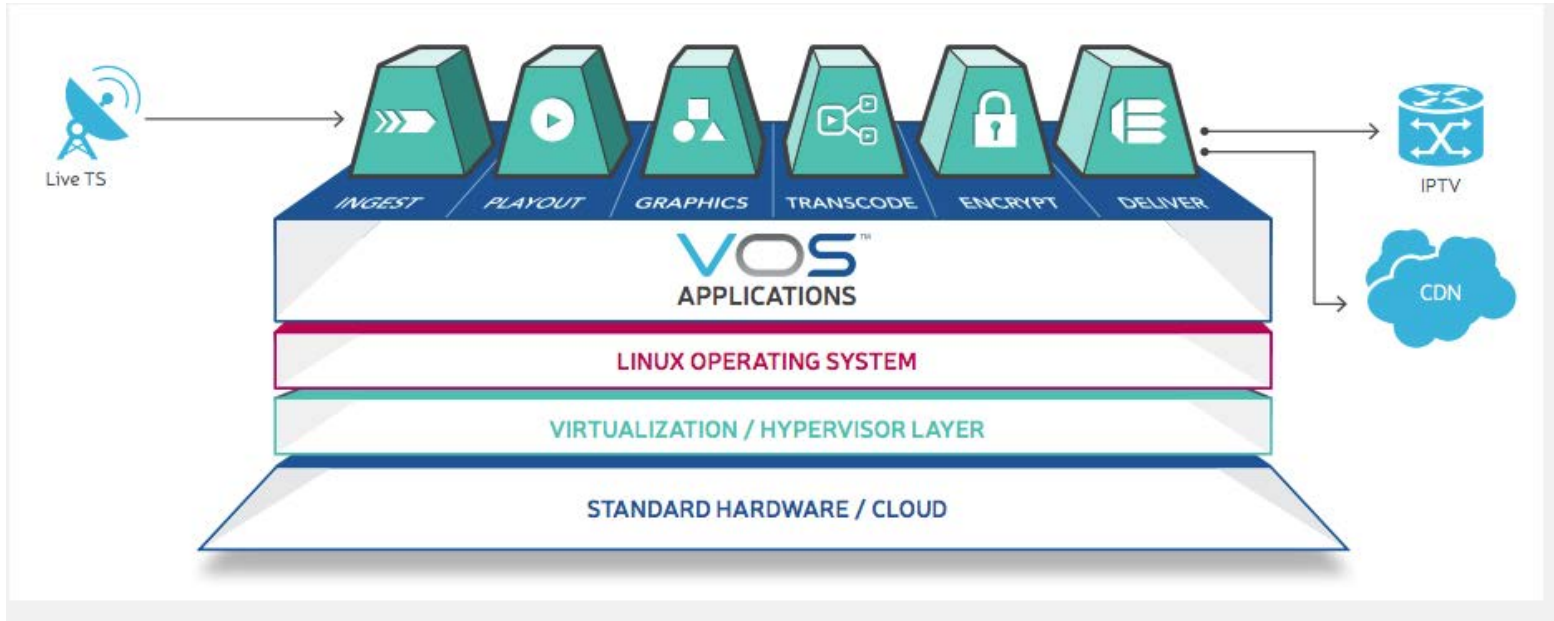
- No pricing info on website

Cloud Transcode: Harmonic VOS Cloud Software



- Licensed software
- Deploy in OpenStack or AWS
- No pricing info on website
- Live and VOD

Cloud Transcode: Harmonic VOS Cloud Software



- Licensed software
- Deploy in OpenStack or AWS
- No pricing info on website
- At Streaming Media West
- Live and VOD

Full Transcode and Package: Telestream Lightspeed Live Stream



- ~\$44,950

Live Contribution

- Send H.265 stream to the cloud for transcoding to multiple streams and transmuxing
 - The benefit of HEVC is better quality and/or reduced outbound bandwidth requirements
- Products (not exhaustive)
 - Harmonic
 - LiveU
 - Teradek
 - Vitec MGW Ace

Cloud Transcode: Harmonic ViBE 4K



- Hardware/VOD
- Needs external packager for HLS
- No pricing info on website

Contribution: LiveU



HEVC Pro Card
(for LU) 600
(Ethernet)



Contribution: Teradek



Cube 705
(Ethernet)



Cube 755
(Ethernet +
Wi-Fi)



Slice 756
(Ethernet +
Wi-Fi)

Vitec MGW Ace



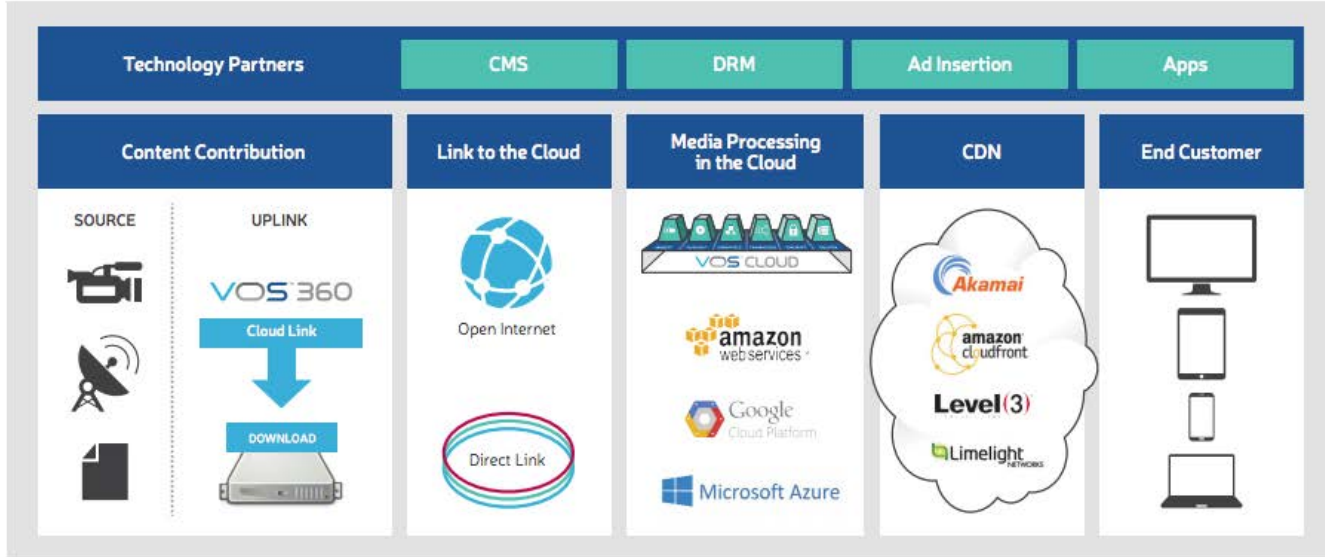
- Hardware appliance

Live Cloud Transcode

- Harmonic VOS 360 cloud service
- Nimble Streamer
- Wowza

Cloud Transcode: Harmonic VOS 360 Service

VOS 360 ECOSYSTEM



- Linux-based software; deploy anywhere

- No pricing info on website
- At Streaming Media West

Nimble Streamer

- How to Create a Live HLS Feed With HEVC
 - http://bit.ly/hevc_hls_nim
- Streaming Media article

The screenshot displays the WMSPanel interface for a Nimble Streamer instance. The main window shows a workflow diagram with two streams: a source stream labeled '/fmp4/stream' and an output stream labeled '/fmp4/hecvc_1080p'. Two dialog boxes are open over the interface:

- EDIT SOURCE STREAM:** This dialog is for configuring the source stream. It has tabs for 'Stream' (selected) and 'File'. The 'Application name' is set to 'fmp4' and the 'Stream name' is 'stream'. The 'Decoder' is set to 'Default' and 'Threads' is set to 1.
- EDIT OUTPUT STREAM:** This dialog is for configuring the output stream. The 'Application name' is 'fmp4' and the 'Stream name' is 'stream_1080p'. The 'Encoder' is 'h265', 'Key frame alignment' is 'source', and 'Codec' is 'h265'. The 'preset' is 'default' and the 'bitrate' is '2500'. There is a 'tune, profile, etc...' field and a link to 'How to set bitrate'. The 'Experts setup' button is also visible.

The background interface includes a top navigation bar with 'Dashboard', 'Reporting', 'Servers', 'Wowza Nimble Streamer', 'Transcoders', 'Control', 'Monitoring', and 'Download'. The user 'janzen@gmail.com' is logged in. A 'Save' button is visible on the right side of the 'EDIT OUTPUT STREAM' dialog.

Wowza

- Can transcode to HEVC/not yet compliant with HLS spec
 - No CMAF yet, are in transition

HEVC, HLS, and Live Production: A Wowza Interview



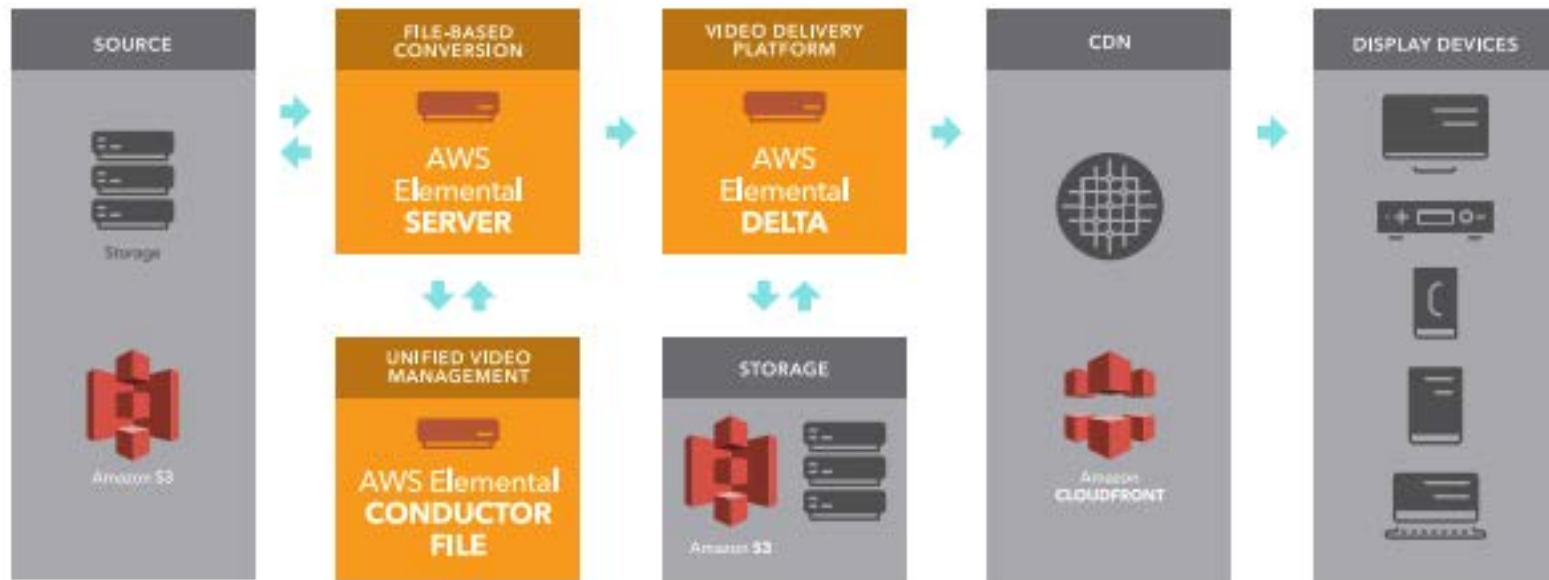
Wowza VP of Engineering Barry Owen

http://bit.ly/wz_hls

VOD

- Appliance
- Software
- Cloud

Appliance: AWS Elemental Server

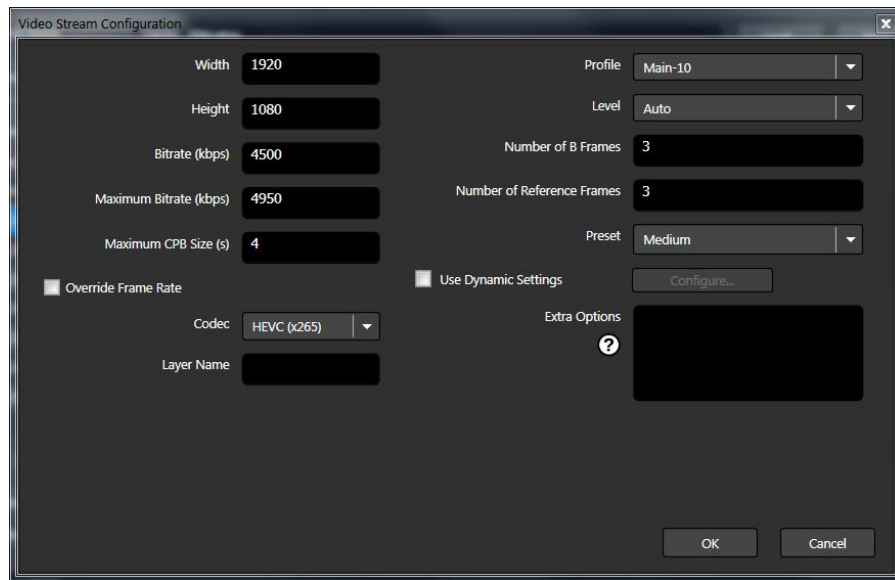


- Linux-based software; deploy anywhere

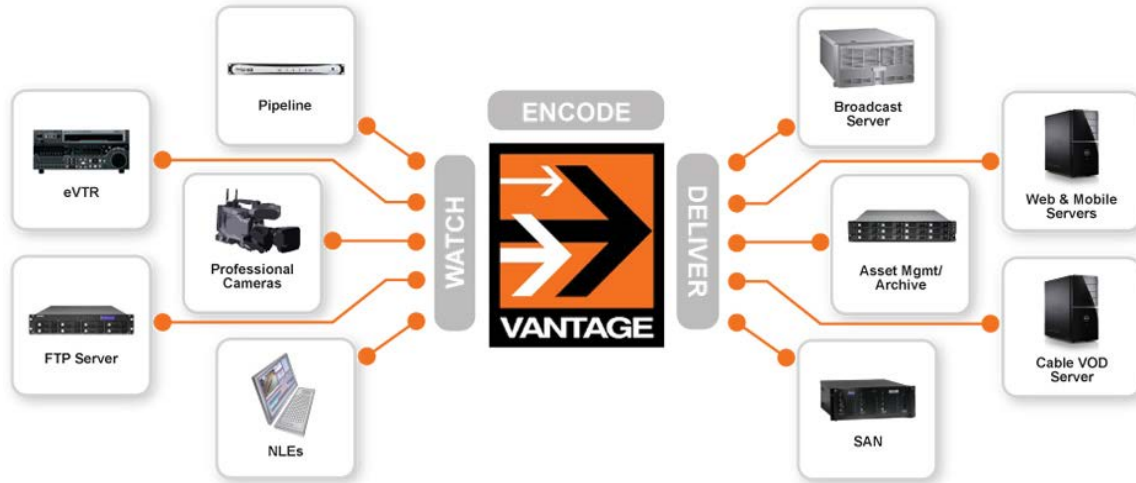
- No pricing info on website
- At Streaming Media West

Capella Systems Cambria FTC Encoder

- Standalone/Cluster
- Windows-based; install anywhere
- www.capellasystems.net

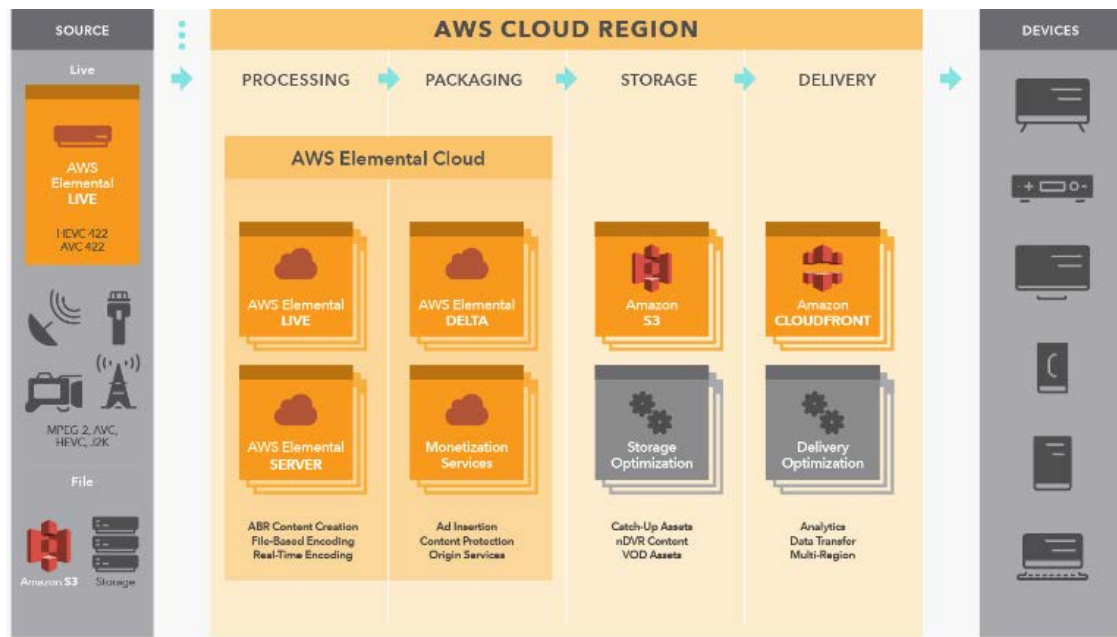


Software: Vantage Media Processing Platform



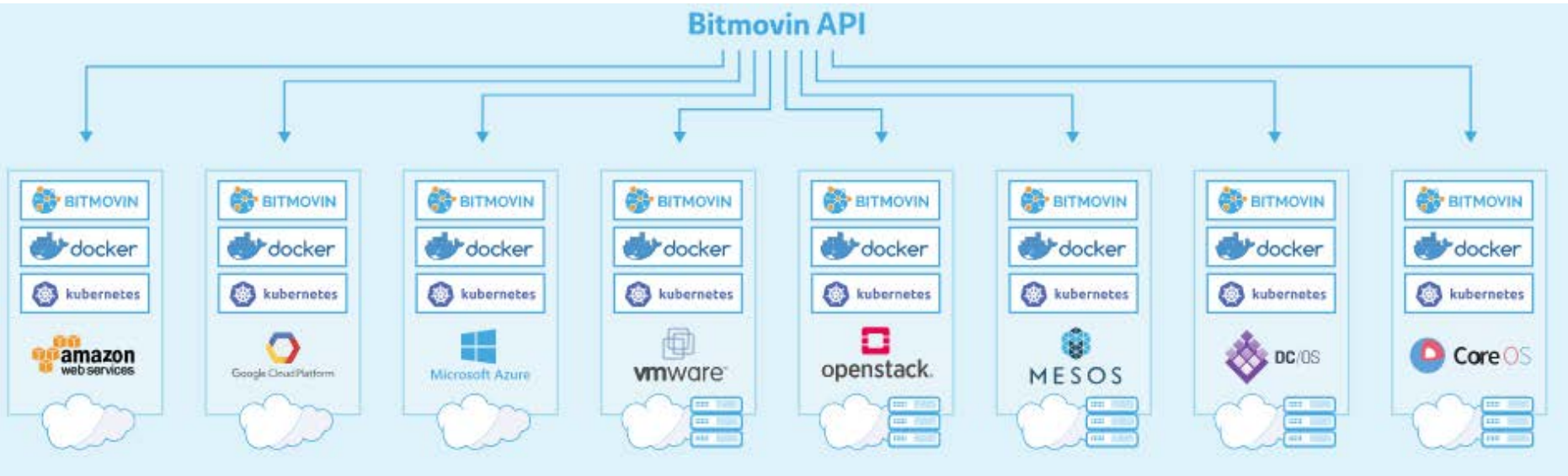
- Can run on servers or on public and private virtualized infrastructures
- At show

Cloud: AWS Elemental Cloud



- True cloud-based product; extensible with other products
- No pricing info on website

Software/Cloud: Bitmovin Video Encoding



- Available as a SaaS offering or for internal deployment
- No pricing info on website

Cloud: Hybrik Cloud

Hybrik
Jan Ozer

Dashboard
Storage
Media Analyzer
Tasks
Jobs
Watch Folders
Presets
Transcode Presets

Presets / Transcode

Action

Name
System Presets
Broadcast
Distribution
Apple HLS
MPEG-DASH
Microsoft Smooth Streaming
WebM
mp4

Preset Details Preset History

Copyright Hybrik Inc © 2014-2017

ALL HYBRIK PLANS INCLUDE:

- Dedicated Machines 24/7/365
- Virtual Private Cloud
- Total Control
- Transcoding and QC
- Accelerated Transfers
- Easy-to-Integrate API
- Email and Phone Support
- No Extra Charges — for Anything!

Big
10 Machines
\$1,000/month

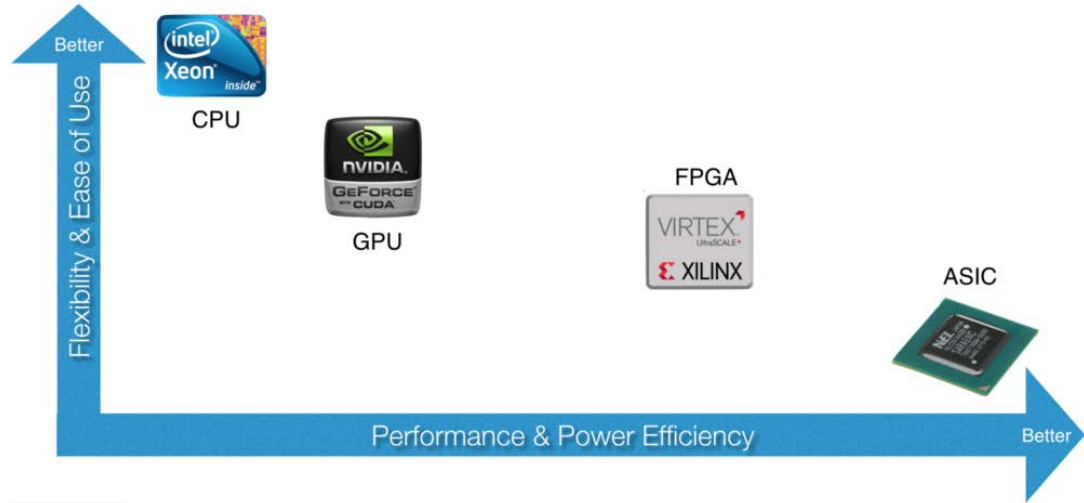
Bigger
100 Machines
\$5,000/month

Biggest
1000 Machines
\$10,000/month

- Currently VOD; moving to live

Field Programmable Gate Array (FPGA)

- NGCodec – FFmpeg plug in that delivers significant acceleration over CPU/GPU
- Up to 30 simultaneous streams
- AWS has cloud machines with FPGAs
 - Cuts cost because fewer machines needed and accelerates encoding
 - Xilinx is at the show along with NGCodec



Other Vendors

- Live

- VOD
 - SDKs
 - Beamr
 - MainConcept
 - Multicoreware