ENCODING LIVE AND VOD FOR HEVC/HLS

A Joint SLC/RealEyes Production
Agenda

- Our assumptions and goals
- Section I: Introduction to HEVC
- Section II: Introduction to HLS
- Section III: Specification overview: HEVC in HLS
- Section IV: Playback performance and ladder composition
- Section V: Producing HEVC/HLS
  - VOD
  - Live
  - Hardware encoding
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
Android Support of HEVC

Android OS supports Main Profile Level 3 (Level 4.1 for Android TV)

- Level of HEVC support in Android OS is relatively low
- Likely supported by hardware decode in many devices in US and Europe (but perhaps not in third world countries).

<table>
<thead>
<tr>
<th>Format / Codec</th>
<th>Encoder</th>
<th>Decoder</th>
<th>Details</th>
<th>Supported File Type(s) / Container Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.265 HEVC</td>
<td></td>
<td></td>
<td>Main Profile Level 3 for mobile devices and Main Profile Level 4.1 for Android TV</td>
<td>MP@4 (mp4)</td>
</tr>
</tbody>
</table>

- Level 3 = 540p
- Apple supports Level 5 (4K/30p)
HEVC and Content 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
- Technicolor – also owns HEVC IP, but seems focused on larger entities
- Many others undeclared
- Content royalties seem unlikely, but are certainly possible
HEVC Codecs

• Because HEVC is a standard, there are many HEVC codecs
• x265 is the open-source HEVC encoder included with FFmpeg
  • Very accessible; may not be the highest quality
• Many others (Beamr, MainConcept) not included in Moscow State University Study

Moscow State University Codec Rankings

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 - ditto

• 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
  - HLS Authoring spec:
    - “1.6. Profile, Level, and Tier for HEVC MUST be less than or equal to Main10 Profile, Level 5.0, High.”

https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding
Main or Main10?

- Apple supports both
- Required for HDR
- With 8-bit input, Main 10 has a very slight quality advantage
  - Encode with Main10 when encoding for HLS
- Standard FFmpeg build may not support Main10
  - My have to compile your own
HEVC Levels

- Set constraints within profiles
- Enable compatibility with lower power devices
- Apple spec:
  - No higher than Main10 Profile, Level 5.0, High Tier (which seems limited to 4K30p)
  - Encoding ladder (as you’ll see) says same as source for HEVC (HDR is 30p limit)
  - Unresolved issue at this point
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

```plaintext
<table>
<thead>
<tr>
<th>preset</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctk</td>
<td>32</td>
<td>32</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>min-cu-size</td>
<td>16</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>bframes</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>b-adapt</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>rc-lookahead</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td>lookahead-slices</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scene-cut</td>
<td>0</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>ref</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>limit-refs</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>me</td>
<td>dia</td>
<td>hex</td>
<td>hex</td>
<td>hex</td>
<td>hex</td>
<td>hex</td>
<td>star</td>
<td>star</td>
<td>star</td>
<td>star</td>
</tr>
<tr>
<td>merge</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>92</td>
</tr>
<tr>
<td>subme</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>rect</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>amp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>limit-modes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>max-merge</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>early-skip</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>recursion-skip</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fast-intra</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>b-intra</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>sao</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>sign/side</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>weighttp</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>weighttb</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>acq-mode</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>cuTree</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rdLevel</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>rdoq-level</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>tu-intra</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>tu-inter</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>limit-tu</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```
## x265 Presets

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

- **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

97.93% total quality (PSNR) ~ 50% Medium encoding time
99.47% total quality (PSNR) ~3x Medium encoding time
Section II: Playback Performance

• How HEVC compares to H.264
  • The assumption:
    • HEVC will work better on newer hardware that supports HW acceleration
    • HEVC will have good quality with lower CPU consumption when HW acceleration is used
  • The caveat:
    • Many platforms still don’t support HEVC: [http://caniuse.com/#search=h.265](http://caniuse.com/#search=h.265)
Section II: 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 (rez/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

• 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
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
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
Section III: 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


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

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Level *</th>
<th>Key Frame</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>145</td>
<td>416 x 234</td>
<td>≤ 30 fps</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>365</td>
<td>480 x 270</td>
<td>≤ 30 fps</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>730</td>
<td>640 x 360</td>
<td>≤ 30 fps</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>1100</td>
<td>768 x 432</td>
<td>≤ 30 fps</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2000</td>
<td>960 x 540</td>
<td>source</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3000</td>
<td>1280 x 720</td>
<td>source</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>4500</td>
<td>1280 x 720</td>
<td>source</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>6000</td>
<td>1920 x 1080</td>
<td>source</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>7800</td>
<td>1920 x 1080</td>
<td>source</td>
<td>High</td>
<td>4.2</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

* Level: Should not use a higher level than required for content resolution and frame rate
**H264 Encoding Ladder – I-Frame/Trick Play**

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Key Frame</th>
<th>Profile</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>480 x 270</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>90</td>
<td>640 x 360</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>250</td>
<td>960 x 540</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>375</td>
<td>1280 x 720</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>600</td>
<td>1920 x 1080</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
</tbody>
</table>
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
• Apple HLS sample stream @ 60 fps (but peak at 1080p)
• Encoding ladder says 30 fps
• Must be fragmented MP4
## HEVC Encoding Ladder - Content

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Level *</th>
<th>Key Frame</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>145</td>
<td>416 x 234</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>300</td>
<td>480 x 270</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>660</td>
<td>640 x 360</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>990</td>
<td>768 x 432</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>1700</td>
<td>960 x 540</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2400</td>
<td>1280 x 720</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3200</td>
<td>1280 x 720</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>4500</td>
<td>1920 x 1080</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>5800</td>
<td>1920 x 1080</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>8100</td>
<td>2566x1440</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>11600</td>
<td>3840x2160</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>16800</td>
<td>3840x2160</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

* Level: Should not use a higher level than required for content resolution and frame rate
## HEVC Encoding Ladder – I-Frame/Trick Play

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Key Frame</th>
<th>Profile</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>480 x 270</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>80</td>
<td>640 x 360</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>210</td>
<td>960 x 540</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>300</td>
<td>1280 x 720</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>525</td>
<td>1920 x 1080</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Level *</th>
<th>Key Frame</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>416 x 234</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>360</td>
<td>480 x 270</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>800</td>
<td>640 x 360</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>1200</td>
<td>768 x 432</td>
<td>≤ 30 fps</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2050</td>
<td>960 x 540</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2900</td>
<td>1280 x 720</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3850</td>
<td>1280 x 720</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>5400</td>
<td>1920 x 1080</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>7000</td>
<td>1920 x 1080</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>9700</td>
<td>2560x1440</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>13900</td>
<td>3840x2160</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>20000</td>
<td>3840x2160</td>
<td>30</td>
<td>Main 10</td>
<td>5.0</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

* Level: Should not use a higher level than required for content resolution and frame rate
HDR Encoding Ladder – I-Frame/Trick Play

<table>
<thead>
<tr>
<th>Data Rate</th>
<th>Rez</th>
<th>Frame rate</th>
<th>Profile</th>
<th>Key Frame</th>
<th>Profile</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>480 x 270</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>640 x 360</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>250</td>
<td>960 x 540</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>360</td>
<td>1280 x 720</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>650</td>
<td>1920 x 1080</td>
<td>1 fps</td>
<td>High</td>
<td>1</td>
<td>High</td>
<td>1</td>
</tr>
</tbody>
</table>

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

- 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

- 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

- I-frame variants in HEVC/H264 formats
- Dolby obviously not required

https://developer.apple.com/streaming/examples/
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"
IV: Playback Performance and Ladder Creation

- Hybrid and low hybrid
- Tests and results
- Conclusions
Created Two Encoding Ladders for Testing

- “Hybrid”
  - Contained all rungs of recommended H.264 and HEVC ladders
Created Two Encoding Ladders for Testing

- **“Hybrid”**
  - Contained all rungs of recommended H.264 and HEVC ladders
- **“Low-Hybrid”**
  - Sub 720p rungs in H.264
  - 720p and higher rungs in HEVC
Burned File Configuration into Files

- Used FFmpeg text filter to burn rez/codec/data rate info into file
Please Help Me Test HEVC Playback in HLS
Results

- 43 desktop
- 19 mobile
What Did We Learn

• Generally good 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 MacBookPro with 1800 vertical rez retrieved 4K file
  • Otherwise, followed this rule
  • 4K doesn’t get retrieved all that often
    • Average bandwidth when retrieving 4K was 580 Mbps
    • Lowest was 64 Mbps for 16.8 Mbps stream
    • Many devices with very high bandwidth and necessary resolution could not play
    • Apple looking into this as potential “bug”
Does Ladder Composition Make a Difference?

• Maybe
• There were several instances where the result between hybrid and low hybrid differed
  • In all but one instance, the low-hybrid experience was worse
    • Either H.264 instead of HEVC
    • Lower data rate/resolution
• Safest approach appears to be two complete ladders
  • Obviously, also the most expensive
What Know About Switching?

• Ask Apple – two streams in ladder; which does player select?
  
  1080p – HEVC – 2 Mbps
  720p – H.264 – 2.5 Mbps

• Their switching logic is in transition but it “knows” that 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
  • 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


- 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
  - Note “pass” configuration - not like H.264

- Preset, an (audio no), format, and Null outside of this structure

- Scaling commands outside of –x265-params structure
FFmpeg Learning Resources

• Includes H.264/H.265
  • Full documentation of Bento4
  • No cloud stuff
• T103. HOW-TO: Building A More Robust Cloud Encoder With FFmpeg & More
  • Tuesday, November 13: 1:45 p.m. - 2:30 p.m.

Creating HEVC in FFmpeg w/ HW Acceleration

• Use the -hwaccel codecs

• HW Decoders have traditionally been okay but SW as good

• HW Encoders have always gotten less quality at higher BR

• NVIDIA NVENC changes this
Previous Issues in HW Encoding

- HW Decoders great
- HW Encoders not so great
- HW Decoders couldn’t easily pass data to SW Encoders
- Reduced general usefulness of HW decoding at all
- Industry needed a full HW decode and encode solution
What’s The Beef?

- NVENC + NVDEC first true HW pipeline solution for all
- HW decode to HW memory surface
- FFmpeg video filters using HW memory surface benefits
- HW encode from HW memory surface
- Multi-fold reduction in stream copying in FFmpeg
What’s The Beef?

- Allows for more real-time video manipulation
- GPU powered real-time augmented, interactive VR
- FFmpeg control for ease of development
- Only true, easy to access HW pipeline solution
- Encoding structure benefits live use case
How to Use HW Accel

- Need an ND-series Azure VM using Pascal V4 ( P40 )
- Windows 2012 R2 / 2016 or Ubuntu 16.04, RHEL 7.3, 7.4
- Nvidia GPU Driver Extension
- A Pay-As-You-Go or high Azure Subscription
How to Use HW Accel

- FFmpeg today comes w/ hwaccel enabled
- On Linux use nvidia-smi to check GPU Driver install
- Nvidia GPU Driver Extension
- A Pay-As-You-Go or high Azure Subscription
Benefits of Azure w/ Nvidia NVDEC/NVENC

- Same level as today's SW solutions but at higher scale
- Low latency decoding/encoding and videograph filters
- Unique live, low-latency HEVC/H265 solutions
- Share hardware surface memory of GPUs between VMs
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/codec 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
HLS 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

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)
Dual HLS and DASH From fMP4

DEMO
Let’s see this happen
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/
PACKAGERS: SHAKA
Introduction to Shaka

- Shaka has made many performance improvements over the past year.
- Makes it simple to demux audio and captions
- Simple DRM integration
- The only known open source packager to have Ad CUE capabilities
Introduction to Shaka

- packager
  'in=/media/input.mp4,stream=audio,segment_template=audio/$Number$.ts,playlist_name=audio/stream.m3u8,hls_group_id=audio,hls_name=ENGLISH'
  'in=/media/input.mp4,stream=video,segment_template=video/$Number$.ts,playlist_name=video/stream.m3u8,iframe_playlist_name=video/iframe.m3u8' --enable_widevine_encryption --key_server_url https://license.uat.widevine.com/cenc/getcontentkey/widevine_test --content_id 7465737420636f6e74656e74206964 --signer widevine_test --aes_signing_key
  1ae8ccd0e7985cc0b6203a55855a1034afc252980e970ca90e5202689f947ab9 --aes_signing_iv d58ce954203b7c9a9a9d467f59839249 --hls_master_playlist_output master.m3u8
BUILDING A FULL SOLUTION: CLOUD/ON-PREM
VOD: Server-based HEVC/HLS Asset Generation

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

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


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)
  • Adapters (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
SAMPLE CODE

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

```javascript
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}`;
input.inputOptions.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 Transpareently
  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/

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](https://aws.amazon.com/autoscaling)

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

• Wowza

• 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

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, at least for initial stream (contribution)
FFmpeg Live

```bash
ffmpeg -re -i input \
-y -c:v libx265 -preset fast \
-x265-params profile=main:keyint=48:bitrate=4500:vbv-maxrate=4500:vbv-bufsize=9000 \
-c:a aac -b:a 128k -ac 2 -ar 48000 output_1080p.mp4 \
\n-y -c:v libx265 -s 1280x720 -preset fast \
-x265-params profile=main:keyint=48:bitrate=2500:vbv-maxrate=2500:vbv-bufsize=5000 \
-c:a aac -b:a 128k -ac 2 -ar 48000 output_720p.mp4 \
\n-y -c:v libx265 -s 640x360 -preset fast \
-x265-params profile=main:keyint=48:bitrate=1000:vbv-maxrate=1000:vbv-bufsize=2000 \
-c:a aac -b:a 128k -ac 2 -ar 48000 output_360p.mp4
```

- Input is typically from a capture device or incoming stream
- Outputs will be to server addresses
- Easiest if you can encode complete ladder on a single instance
  - Otherwise, split all rungs over multiple computers
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
Resources

• Slides: http://bit.ly/2gwIYs5
Third Party Alternatives

- Live
  - Full transcode and package
  - Contribution
  - Cloud transcode

- VOD
  - Appliance
  - Software
  - Cloud
Live: Full Transcode and Package

- DVEO Gearbox265
- Elemental Live
- Harmonic Electra XT
- Harmonic VOS Cloud Software
- Telestream Vantage Lightspeed
Full Transcode and Package: DVEO Gearbox265

- Hardware appliance
- No pricing info on website
  - At Streaming Media West
Full Transcode and Package: Elemental Live

- Linux-based software; deploy anywhere
- No pricing info on website
  - At Streaming Media West
Full Transcode and Package: Harmonic Electra XT, X2, X2S, VS

- Linux-based software; deploy anywhere
- No pricing info on website
- At Streaming Media West
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

- Linux-based software; deploy anywhere
- No pricing info on website
- At Streaming Media West
Live Contribution

- Harmonic
- LiveU
- Teradek
Cloud Transcode: Harmonic ViBE 4K

- Hardware/VOD
- Needs external packager for HLS
- No pricing info on website
- At Streaming Media Westn
Contribution: LiveU

- **HEVC Pro Card**
  - (for LU) 600
  - $2,790 (Ethernet)

- **Cube 755**
  - $2,990 (Ethernet + Wi-Fi)

- **Slice 756**
  - $3,990 (Ethernet + Wi-Fi)
Contribution: Teradek

Cube 705
$2,790
(Ethernet)

Cube 755
$2,990
(Ethernet + Wi-Fi)

Slice 756
$3,990
(Ethernet + Wi-Fi)
Live Cloud Transcode

- Harmonic VOS 360 cloud service
- Wowza
Cloud Transcode: Harmonic VOS 360 Service

- Linux-based software; deploy anywhere
- No pricing info on website
  - At Streaming Media West
Wowza

• Can transcode to HEVC/not yet compliant with HLS spec
  • No CMAF yet
  • Here at show; ask when they will have

HEVC, HLS, and Live Production: A Wowza Interview

Wowza VP of Engineering Barry Owen

VOD

• Appliance
• Software
• Cloud
Appliance: AWS Elemental Server

- Linux-based software; deploy anywhere
- No pricing info on website
- At Streaming Media West
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
• At Streaming Media West
Software/Cloud: Bitmovin Video Encoding

- Available as a SaaS offering or for internal deployment
- No pricing info on website
- At Streaming Media West
Cloud: Hybrik Cloud

- Currently VOD; moving to live
- At Streaming Media West
Other Vendors

• Live
  • Contribution
    • Vitec – multiple encoders

• VOD
  • SDKs
    • Beamr
    • MainConcept
    • Multicoreware