HLS & DASH multi codec encoding & packaging

Nick Krzemienski nick@fubo.tv

fuboTV VOD

Approx 40,000 VODs in license window today

Approx 500 1hr long video source files received via aspera daily

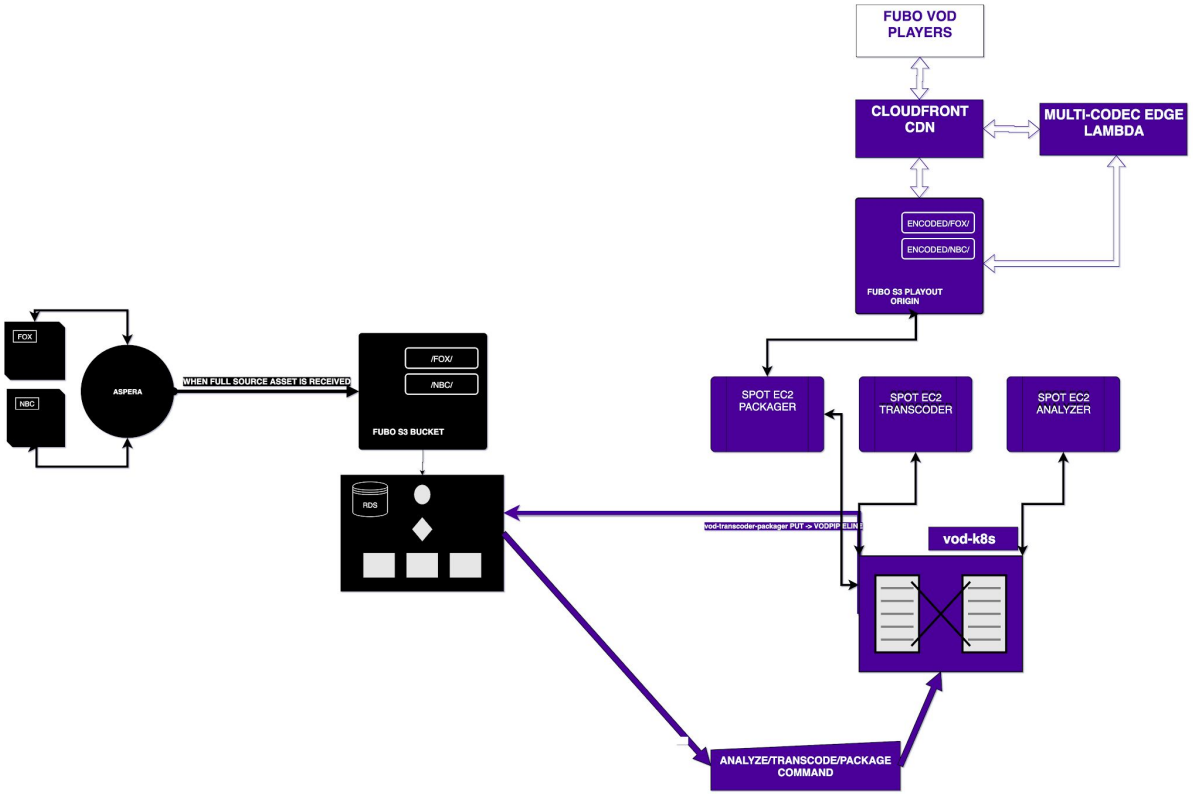
Source files are avg 25gb each

Key takeaways

- What blockers might exist if you wanted to deliver multi codecs
- How you can leverage your edge to help you with those blockers
- A high level understanding of player fragmentations
- A high level understanding of how drm will come into play

BECAUSE OF #EXT-X-TARGETDURATION

- **Benefits we now see (or don’t..)**



- Aspera to S3 for provider source file delivery
- Web server polling source file s3
- Dolby/hybrik for transcoding & packaging on EC2 spot instances via eks and a k8s deployment
- S3 for playout origin
- Cloudfront for playback CDN
- Lambda edge for multi codec origin logic

Players & Devices

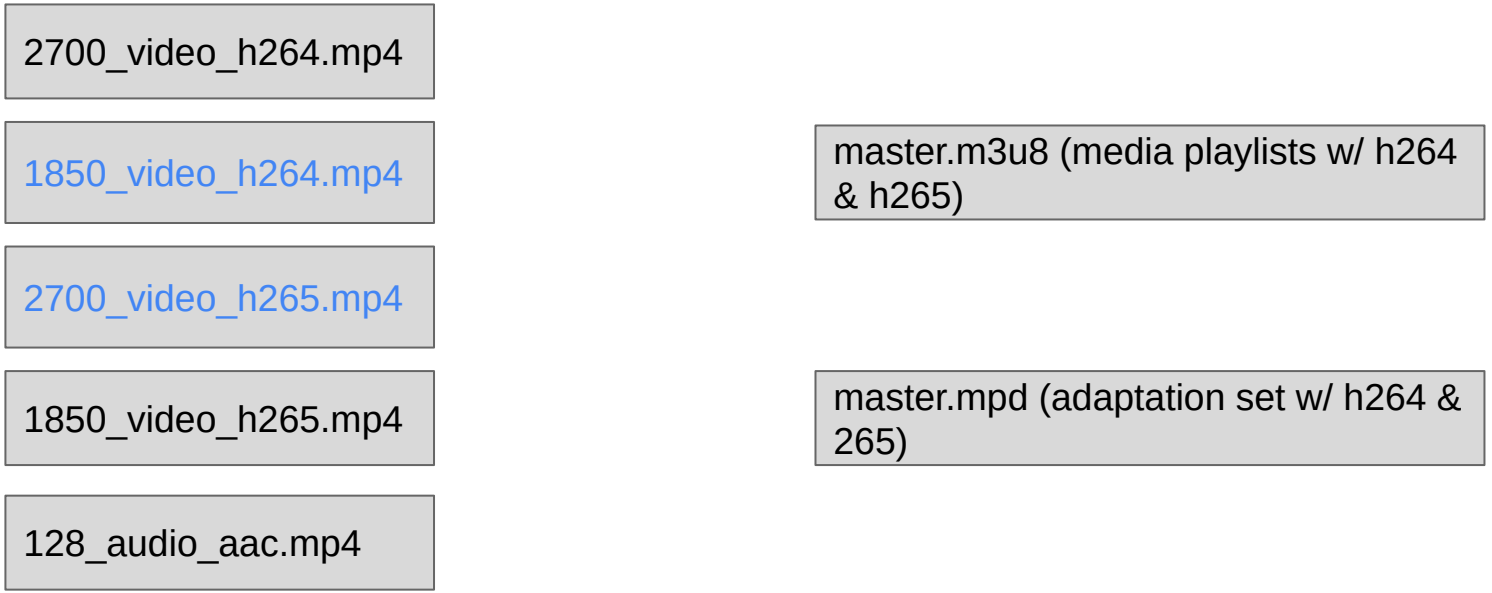
- AVPlayer - ios & tvos
- ExoPlayer - firetv & Android
- Roku native Player - rokuOS
- Bitmovin - web html5
- SmartTV - Tizen
- Chromecast - Shaka player
- Hisense native Player - hisense
- Xbox native player - xbox

What our players currently are delivered

- Native AVPlayer - HLS
- ExoPlayer - DASH
- Roku Player - DASH
- Bitmovin - DASH
- SmartTV - DASH
- Chromecast - DASH
- Hisense - DASH
- XBOX - DASH

An ideal world

1. Transcode one set of fmp4s to both h264 & h265 outputs
2. Package both HLS & DASH from set of transcoded h264 & h265 outputs
3. Players choose what they support



Multi package solution

1. Transcode one set of fmp4s to both h264 & h265 outputs
2. Package HLS with both h264 & h265
3. Package DASH h264 & Package DASH h265

2700_video_h264.mp4

1850_video_h264.mp4

2700_video_h265.mp4

1850_video_h265.mp4

128_audio_aac.mp4

master.m3u8 (media playlists w/ h264 & h265)

master.mpd (adaptation set w/ h264 only)

master-hevc.mpd (adaptation set w/ h265 only)

Player multi-codec playlist playback support

Apple/AVFoundation - good to go w/ hls that has both h264 & h265

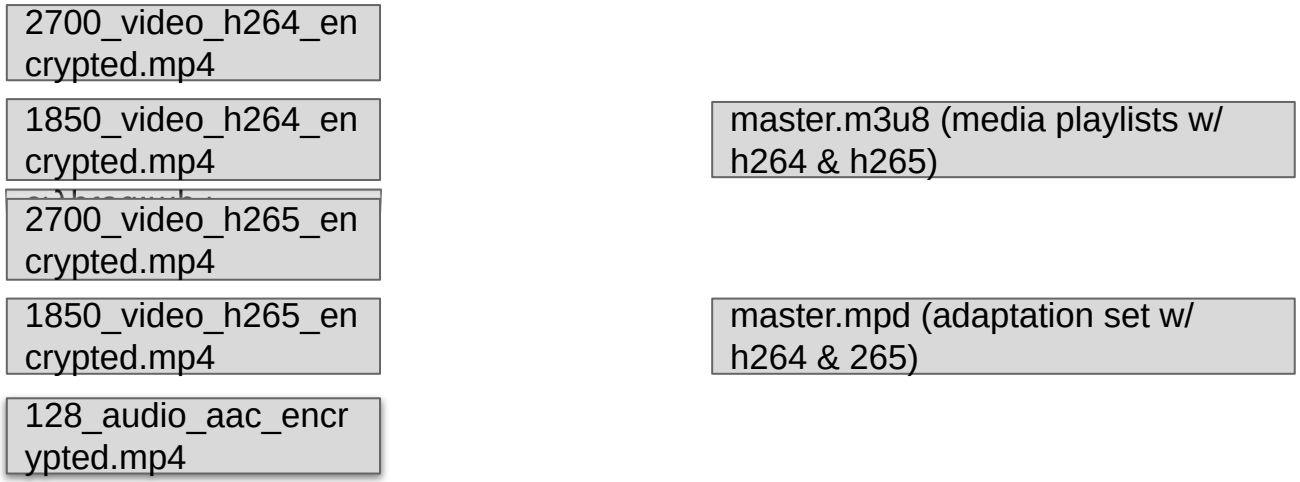
Exoplayer - good to go w/ hls & dash that has both h264 & h265

Roku - fail.... Only capable of playback with single codec playlists h265 only dash/hls or h264 only dash/hls prior to roku (AS OF JANUARY 2020)

With RokuOS as of 9.3 this should be fixed but have not tested

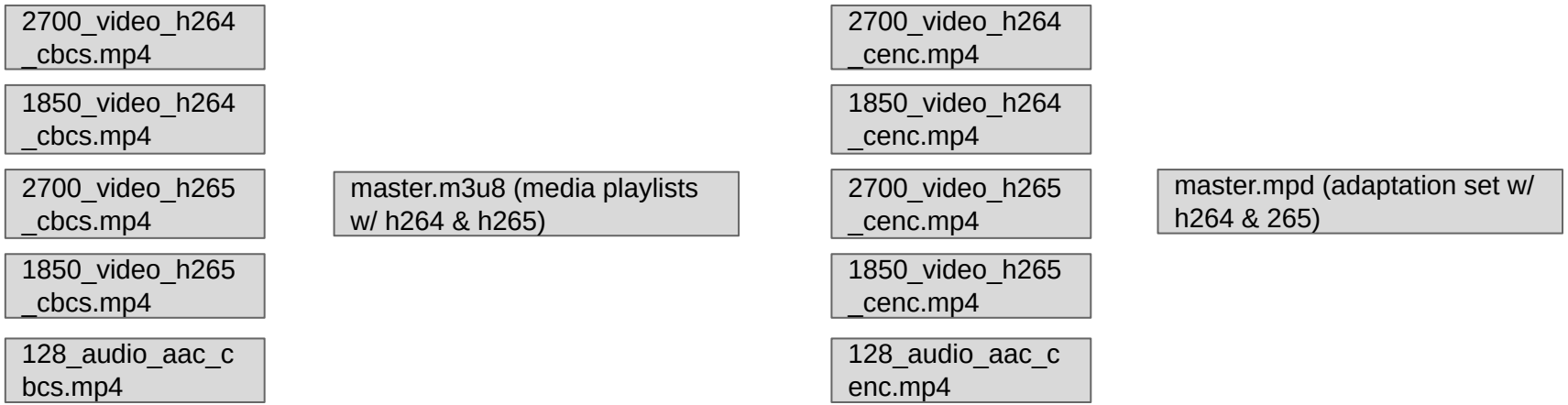
An ideal world w/ drm

1. Transcode one set of fmp4s to both h264 & h265 outputs
2. Package & Encrypt both HLS & Dash from set of transcode h264 & h265 outputs
3. Encryption is done one time for both dash players and hls players (method of encryption)
4. All devices can decrypt all media formats



Common media *almost* format (CMAF)

1. CMAF has different methods of encryption
   a. CENC
   b. CBCS
2. Apple players require cbcs w/ hevc
3. Most other players can also decrypt cenc and hevc



Quick summary

- 2 sets of media for hls and dash if you need to support CENC
- Potentially particular DASH players might not be able to switch between two video codecs

https://www.linkedin.com/pulse/its-cbcsing-time-phil-harrison

Device		CONTAINERS		CODECS			CDM		Manifest Type		DSP			
	Version	CMAF	WebM	AVC	HEVC	AV1	cenc-cbcs	cenc-ctr	HLS	DASH	WideVine	PlayReady	FairPlay	Notes
Chrome	66	YES	YES	YES				YES	YES	YES	YES			
Chrome	67	YES	YES	YES				YES	YES	YES	YES			
Chrome	68	YES	YES	YES		YES	YES	YES	YES	YES	YES			
Safari	10	YES	YES	YES			YES		YES	YES			YES	
Safari	11	YES	YES	YES	YES		YES		YES				YES	
Safari	12	YES	YES	YES	YES		YES		YES				YES	
Firefox	59	YES	YES	YES				YES	YES	YES	YES			
Firefox	60	YES	YES	YES		YES	YES	YES	YES	YES	YES			
Internet Explorer	11		YES	YES				YES		YES				Windows 10 only
Edge	17	YES		YES	YES			YES		YES		YES		Windows 10 only
Edge	18	YES		YES	YES		YES	YES		YES			YES	Windows 10 only
iOS/tvOS	10	YES		YES	YES		YES		YES				YES	Windows 10 only
iOS/tvOS	11	YES		YES	YES		YES		YES				YES	
iOS/tvOS	12	YES		YES	YES		YES		YES				YES	
Android	7.1	YES	YES	YES	YES		YES	YES	YES	YES	YES			Android 7.1 only for CBCS
Android	8	YES	YES	YES	YES		YES	YES	YES	YES	YES			
Android	9	YES	YES	YES	YES		YES	YES	YES	YES	YES			
Roku	8.1	YES	YES	YES	YES			YES	YES	YES	YES			PlayReady 3.0
XBOX ONE			YES	YES				YES		YES		YES		
PS4			YES	YES				YES		YES		YES		PS4 Pro has PR 3.0
Tizen	15		YES	YES	YES			YES	YES	YES	YES	YES		
Tizen	16	YES	YES	YES	YES			YES	YES	YES	YES	YES		
WebOS	16	YES	YES	YES	YES			YES	YES	YES	YES	YES		
WebOS	17	YES	YES	YES	YES			YES	YES	YES	YES	YES		
Chromecast 1	1.3	YES	YES	YES				YES	YES	YES	YES	YES		
Chromecast 2	1.3	YES	YES	YES				YES	YES	YES	YES	YES		
Chromecast Ultra	1.3	YES	YES	YES	YES			YES	YES	YES	YES	YES		

Edge logic (one header to rule them all)

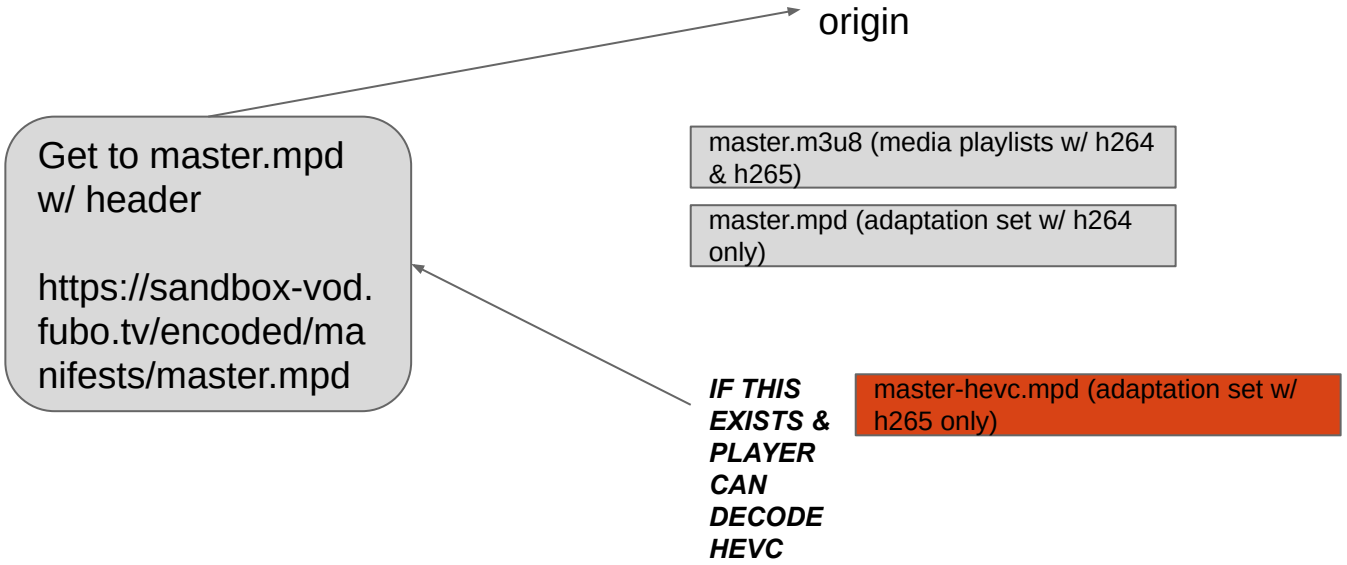
- Players add header with their supported decoding capabilities

X-Supported-Codecs-List: hevc,avc,vp9

- The lack of the above header is treated as part of the logic
- Need for backwards compatibility with previous single codec assets that are in prod
- Need for a roll out over time, only new encodes will have hevc, we use the origin as the source of truth as to whether this is a hevc and avc asset

```
Function FourKCheck() as Object
    deviceInfo = CreateObject("roDeviceInfo")
    hdmiStatus = CreateObject("roHdmiStatus")
    displayProperties = deviceInfo.GetDisplayProperties()
    check = {
        supported    : True,
        videoCodecs : [],
        hdrModes : []
    }

    ' Output check
    video_mode = deviceInfo.GetVideoMode()
    if (Left(video_mode, 5) <> "2160p") then check.supported = False

    ' Check hdmi support
    if hdmiStatus.IsHdcpActive("2.2") <> true then check.supported = False

    ' Codecs
    if check.supported = True
        ' Video
        videoCodecs = check.videoCodecs
        hdrModes = check.hdrModes

        ' HEVC
        hevcVideo = { Codec: "hevc", Profile: "main", Level: "5.1" }

        if deviceInfo.CanDecodeVideo(hevcVideo).result = True then videoCodecs.push("hevc")

        ' vp9
        vp9Video = { Codec: "vp9", Profile: "profile 0" }
        if deviceInfo.CanDecodeVideo(vp9Video).result = true then videoCodecs.push("vp9")

        ' AVC
        avcVideo = { Codec: "mpeg4 avc", Profile: "main" }
        if deviceInfo.CanDecodeVideo(avcVideo).result = true then videoCodecs.push("avc")

        ' HDR Modes
        ' Hdr10
        if displayProperties.Hdr10 = True then hdrModes.push("hdr10")

        ' Dolby
        if displayProperties.DolbyVision = True then hdrModes.push("dolbyVision")
    end if

    return check
End Function
```

Multi package with origin logic flow



https://gist.github.com/krzemienski/47c79f0a5f94b9e5e7789742144b67a0

```
'use strict';

const https = require('https');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const origin = event.Records[0].cf.request.origin;
    const uri = request.uri;
    const headers = request.headers;

    console.log('Request: ', request);
    console.log('Requested URI: ', uri);

    console.log('Requested origin HEADERS: ', headers);

    if (headers['x-supported-codecs-list']
        && headers['x-supported-codecs-list'].length > 0
        && headers['x-supported-codecs-list'][0].value.includes("hevc")) {

        var masterUri = request.uri;

        const cache_host =  'https://d1l6lr3hcvwkt7.cloudfront.net';
        const cacheUrl =  cache_host + masterUri.replace('master.mpd', 'master-hevc.mpd');

        https.get(cacheUrl, (res) => {
            if (res.statusCode === 200) {
                request.uri = masterUri.replace('master.mpd', 'master-hevc.mpd');

                console.log(`Request uri set to "${request.uri}"`);

                callback(null, request);
            } else {
                callback(null, request);
            }
        }).on("error", (err) => {
            console.log("Error: " + err.message);
            callback(null, request);
        });
    } else {
       callback(null, request);
    }
};
```

Packaging both HLS & DASH w/ shaka & bento4

https://gist.github.com/krzemienski/59ada206f0763db17bf33f5f8702eeb7

8/10 playback bugs in production are relevant to drm and packaging of hls and dash

- Packaging done in a container that takes an s3 directory of transcoded bitrates and codecs as input
- Leveraged open source frameworks shaka packager from google & bento4

Production mishap...

- CCs & Subtitles all of sudden don’t work anymore on a good amount of players
- We are now using fmp4s instead of mpeg2
- Certain players particularly Shaka & Apple don’t pick up embeds on fmp4s
- Suggest a workflow where you normalize all subs and CCs to .vtt or similar

Investing in your vod multi codec pipeline

- Vod2Live (shut down live encoders and have it point to a vod for repetitive content)
- Potentially deliver SDR in an HDR container
- Not limited to HEVC all of this logic is easily replicated and scaled to other ***fancy 3 letter acronyms***

I work for tshirts (hoodies accepted as well)

https://awesome.video

Please reach out, would love to help!

nick@fubo.tv

@nick in https://video-dev.slack.com

Questions?