Roku

Streaming Player Encoding Guide

Roku Streaming Player Version 4.1 12/21/2011

Table of Contents

1.0 Introduction

Roku has been pleased with the growth in the number of channels so far and we are excited about further accelerating the development of more channels. We hope this guide will help in this endeavor.

There are two primary methods of encoding content for the Roku Streaming Player. The first is our traditional multi-file support. The Roku Streaming Player supports the encoding of different bitrates for a piece of content and the passing of urls referencing the different mpeg4 files for each bitrate to the video player. Whenever the video player detects an underrun, it can switch to a lower bitrate encoding and continue playing at the current timestamp after buffering the required amount of data for that bitrate.

The second encoding method enables a more adapative switching of bitrate selection. This encoding method is called HTTP Live Streaming (HLS). HLS can proactively change the bitrate it plays on any "segment boundary" (usually at 10 second intervals) without an underrun or buffering event. HLS has seen rapid growth recently and is the only way to deliver live content on the Roku Streaming Player. Although HLS is the cutting edge, this encoding guide will provide pointers and working recipes for successful implementations. More important than the actual recipe, we'll try to give you a sense of tradeoffs available and where different types of content might fit.

2.0 Planning your Encoding

Before starting your encodings, you will need to understand the constraints of the Roku Streaming Player and your source material.

2.1 Roku Streaming Player Requirements

Movie source content consisting of frames is referred to as progressive. Source content from analog video cameras or old (non-HD) TV broadcasts is referred to as interlaced. Interlaced content consists of two fields on odd and even scan lines that if combined would produce a full resolution frame. The fields are displayed alternatively at a field rate that is twice what the full frame rate would be.

• **Standard Film**	**24 fields/sec (progressive)**
• **NTSC Video**	**60 fields/sec (interlaced)**

Note that if progressive there is a one to one ratio if fields to frames, but if interlaced there is a two to one ratio of fields to frames.

If your source material comes from DVD or broadcast/cable/satellite TV, it will conform to one of two standards: NTSC for North America and Japan, or PAL for Europe, etc. The Roku Streaming Player will only play progressive encodings at frame rates of 23.976 from native film or 29.97 from interlaced source video. If your source content is PAL video or some Computer Graphics that is a factor of 25 frames/sec or any source other than 24, 30, 48, or 60 frames/sec, the video frame rate must be converted as part of the encoding process.

The process of converting a film to video is called "telecine", and requires a method of dealing with the different frame rates. One popular form of telecine is NTSC 3:2 pull-down telecine. When extracting NTSC video fields from standard film frames, 3:2 pull-down alternates using one standard film frame for three NTSC video fields followed by one frame for two fields. This results in a field rate 2.5 times the original frame rate. The result is also slowed down very slightly from 60 fields per second to 60000/1001 fields per second to maintain NT SC field rate. There are other ways of extracting video fields from standard film, but they all introduce either a speedup of the video or a different ratio of frames to fields. As a general rule, DVD movie content is usually progressive at 23.976 fps. DVD TV series content is usually only interlaced, not telecined at 29.97 fps. To further complicate things, sometimes you will run into DVDs that are a mixture of progressive and interlaced. There are even some DVDs that have been “hard telecined”, meaning that the odd scan lines are used to generate both fields. This effectively halves the resolution of the encoding.

Please refer to the Wikipedia entry: http://en.wikipedia.org/wiki/Telecine for further explanation of telecine methods. Note that even 23.976 fps and 29.97 fps are approximations of the actual 24000/1001 or 30000/1001 fps used by NTSC video and the Roku box.

Info: The reason for the rational numbers of fields or frames per second is that the bandwidth of a field per second was dedicated in NTSC video broadcasts to provide color sync information in a backwards compatible way with legacy black and white video fields. When referring to progressive video frame rates, 24, 23.976, and 24000/1001 are all synonyms for the actual 24000/1001 frame rate.

The bottom line is that it is best to use the original film to do the encoding on rather than having to deal with the artifacts like combing (interlacing) artifacts and duplicated or lost frames that are added in the telecine process. In addition to being noticeable to the trained eye, the telecine artifacts degrade the coding efficiency of the encoding. If you have access to mezzanine files or high quality D5 tapes you’ve got access to the best input to your encoding process.

If you don’t have access to the original, and must encode telecine content you will use the “inverse telecine” filters in your encoder to create a progressive encoding at a framerate understood by the Roku box. The encodes for the Roku box will end up at 24 fps for original film content and shot-to-video and mixed material will be de-interlaced to 30 fps for NTSC or 25 fps for PAL.

Constant Bit Rate (CBR) or Variable Bit Rate (VBR)

The Roku box requires that encodings be either Constant Bit Rate (CBR) or Variable Bit Rate (VBR). When you specify a constant bit rate, the encoder will discard as much detail as necessary to stay below the specified bit rate. Take care in choosing “two-pass”, “multi-pass”, Average Bit Rate (ABR), or Constant Quantizer settings in your encoder. Although they improve encoder efficiency, these settings will not always be decoded properly on the Roku box. Those striving for the best encodings at a particular bitrate will choose “two-pass” or “multi-pass” encodings and then actually watch the video on the Roku to assure there are no sustained spikes in bitrate that may cause the Roku to display the “buffering” screen. Those striving for quick encodes that they are confident will play on the Roku will choose “one-pass” or Constant Bit Rate (CBR) encodings. Most content providers end up in the middle of these two extremes with a VBR setting where the variance in bitrate is within about a standard deviation of the mean. Some encoders let you specify the peak bandwidth as well as the average bandwidth. In this case, a VBR setting where the peak is less than 1.5x the average bandwidth makes for a good tradeoff between efficient encodes and confidence in decoding and encoding time.

2.2 Live and VOD

The only way to deliver live content on the Roku is to use HTTP Live Streaming (HLS). Video on Demand (VOD) can be delivered via HLS, mp4, and wmv container files. Note that the Roku HLS implementation will be the only way to deliver playlists with ad insertions in the near future. However, Roku HLS does not currently support BIF files for Trick Mode display.

We will go into more detail about how to encode Live content in Section 3: HTTP Live Streaming.

2.3 Cropping and Scaling

Some source content may have a black border. These borders should be cropped during the encoding process. There is no reason to have black borders in the encoded video, as the Roku will add any necessary borders in order to maintain the aspect ratio when the content is displayed on the television.

The border's sharp edges are very inefficient for the Discrete Cosine Transforms (DCT) used in the H.264 macroblocks. Sharp edges within a block require lots of high frequency information and therefore more bits and negatively affect motion vector calculation. Sometimes there is noise or distortion at the edges of the source content. This should also be cropped out.

2.4 Resolution and Bitrates

A good rule of thumb is that a high quality encode should be about 2.0 Mbps at SD resolution. Some slow moving dramas might get by at 1.5 Mbps for SD, while fast action sports would benefit by going higher than 3.0 Mbps at HD resolution. When encoding for HD, the bit rate must be at least 2.0 Mbps, and will look much better above 2.4 Mbps.

As you sacrifice quality for lower bitrates, we recommend that the lowest bit rate video that is still watchable on the TV screens Roku boxes are plugged into is 394 kbps. When lowering the bit rate, you can bias the quality sacrifice towards resolution or frame rate. The best tradeoff will be source content dependent, but in general fast action like sports would do better with higher frame rates while slow moving scenes like college lectures would benefit more from higher resolution.

Below are some sample Stream Bit Rates for different types of content:

Sports:

#1. 800 kbps
#2. 1386 kbps
#3 1800 kbps
#4. 2400 kbps (HD)
#5. 3200 kbps (HD)

Movie Library:

#1. 664 kbps
#2. 996 kbps
#3. 1320 kbps
#4. 2600 kbps (HD)
#5 3800 kbps (HD 1080p)

College Lectures:

#1. 394 kbps
#2. 664 kbps
#3. 996 kbps
#4.1500 kbps
#5.2000 kbps (HD)

Note that the above guidelines are very dependent on the resources you have at your disposal. For example, in order to support the 5 live HLS bitrates under the sample sports streams above, you would need a high-end encoder like the Inlet 7100 or 8100 and a large pipe to your CDN (Content Delivery Network).

When selecting resolutions, you will generally want 720x480 SD content, 1280x720 HD content, and 1920x1080 1080p HD content. You may decide to set a resolution lower than 720x480 on your low bit rate streams. When selecting these lower resolutions, you should use a resolution that is a multiple of 16 since H.264 uses 16x16 macroblocks (Otherwise you will be wasting bits on resolution that is unused).

2.5 Supported Video Formats

The following is a summary of supported video formats that have been tested and/or are currently in-use. Other formats or encoding may be supported, but should be evaluated on a case by case basis.

	H.264 SD	**H.264 HD**
Aspect Ratio[1]	4:3	16:9
Dimension	Various to 720x480	Various to 1280x720 and 1920x1080 for 1080p
Progressive/Interlaced	Progressive	Progressive
File Format	.mp4 (MPEG-4 Pt 14), .mov .m4v HLS: m3u8 & .ts	.mp4 (MPEG-4 Pt 14), .mov .m4v HLS: m3u8 & .ts
Frame Rate[2]	23.976 fps or 29.97 fps	23.976 fps or 29.97 fps
Color Space	YUV	YUV
Video Codec	H.264/AVC	H.264/AVC
Profile	Main/High	High
Level/Complexity	4.0	4.0
Video Mode	Constrained VBR	Constrained VBR
Average Streaming Video Bitrate[3]	384Kbps - 1.6Mbps	1.6Mbps – 3.2Mbps
Average USB Video Bitrate[3]	384Kbps – 8.0Mbps	384Kbps – 8.0Mbps
Peak Video Bitrate	1.5x average	1.5x average
Key Frame Interval	< 10s	< 10s
DRM	None	None
Audio Codec	AAC LC (CBR), AC3 Passthrough	AAC LC (CBR), AC3 Passthrough
Audio Bit Rate	128-256Kbps	32-256Kbps
Audio Sample Rate	44.1 Khz or 48Khz	44.1 Khz or 48 Khz
Audio Sample Size	16-Bit	16-Bit
Audio Channels	2-Ch Stereo	2-Ch Stereo

Notes:

1) The dimensions vary on a title-by-title basis depending on the source material and the target aspect ratio for the encoding (e.g. 4:3 or 16:9). Content should always be encoded at full width and the height is adjusted. For example, a 1.66 aspect ratio source is encoded as a 720x432 video and displayed as letterboxed for a 4:3 display.
2) The frame rate used for encoding depends on the source material. Film content is generally 23.976 fps, while video content is generally at 29.97.
3) For typical streaming video applications, we recommend a range of ~384Kbps to ~4.0Mbps. For USB playback, we recommend that you stay under 8.0 Mbps. This provides a good balance between quality and support for a wide number of users. In some cases lower and higher bitrates have been used, but this frequently results in poor quality or limits the % of the installed base that can view this encoding.

2.6 Deinterlacing

The Roku box will not play interlaced encodes. If your source content is interlaced, you must deinterlace it during the encoding process. When doing so, please keep in mind that:

- Crop height must be a multiple of 4.
- Any vertical offsets must be a multiple of 4.
- Any vertical scaling must be performed in interlaced mode.
- Postprocessing and denoising filters may not work as expected unless they are used on a single field at a time.

2.7 A/V Sync issues

The Roku box requires that there is an audio stream synced with the video. It does not properly play video that has no audio. If you have an old silent film, you should add a white noise audio track to the video when encoding.

You may also find that some encoding filters, like inverse telecine filters, will drop or duplicate some video frames. You should try to avoid these filters as the lack of a/v sync may be non-uniform, so that there is no audio delay you could set that would fix the entire video.

3.0 MPEG-4 part 14 (.mp4) and MPEG-4 part 10 (H.264)

The Roku supports MPEG-4 part 14, the .mp4 container file format. The video encoding for in this container must be MPEG-4 part 10 AVC Advanced Video Codec, otherwise known as H.264. Although .mp4 files could contain many different types of video encodings, the Roku only supports H.264 encodings.

You are encouraged to encode your content at multiple bit rates as the Roku can take advantage of these different encodings by measuring the bandwidth available to the device and selecting the stream that requires a bandwidth less than the measured bandwidth. Anytime the Roku box needs to buffer more data, it will pick the appropriate stream again. The H.264 encodings are the most efficient encoding on the Roku box. H.264 is used in the .mp4 container as well as HLS. It is often combined with AAC-LC or AC-3 audio in the MP4 container. HLS requires AAC-LC audio. A good general explanation of x264 encoding parameters associated with FFMPeg can be found here: http://rob.opendot.cl/index.php/useful-stuff/ffmpeg-x264-encoding-guide/

3.1 FFMpeg

FFMpeg is a great tool for transcoding content and it is available on virtually every platform. Although FFMpeg is included by default in every major Linux distribution, the default configuration is usually missing libfaac and libx264 due to licensing restrictions. Assuming your organization has obtained necessary licensing rights, you generally will need to rebuild FFMpeg with faac and x264 support.

Here are the steps that worked for us on Fedora:

```
% sudo yumdownloader --source ffmpeg
% sudo rpmbuild --rebuild --with faac ffmpeg-0.5-3.fc11.src.rpm
% cd ~/rpmbuild/RPMS
% sudo rpm -e ffmpeg
% sudo rpm -U --force ffmpeg-libs-0.5-3.fc11.i586.rpm
% sudo rpm -i ffmpeg-0.5-3.fc11.i586.rpm
% sudo rpm -i ffmpeg-devel-0.5-3.fc11.i586.rpm
% sudo rpm -i ffmpeg-debuginfo-0.5-3.fc11.i586.rpm
```

Or you can build from the pristine sources like so:

Get source at: http://ffmpeg.org/download

```
% configure --enable-gpl --enable-nonfree --enable-pthreads \
 --enable-libfaac --enable-libfaad \
 --enable-libmp3lame --enable-libx264

% configure --prefix=/usr --bindir=/usr/bin \
--datadir=/usr/share/ffmpeg \
--incdir=/usr/include/ffmpeg --libdir=/usr/lib \
--mandir=/usr/share/man --arch=i586 \
--extra-cflags='-O2 -g -pipe -Wall \
-Wp,-D_FORTIFY_SOURCE=2 -fexceptions -fstack-protector \
--param=ssp-buffer-size=4 -m32 -march=i586 \
-mtune=generic-fasynchronous-unwind-tables' \
--extra-version=rpmfusion --enable-bzlib --enable-libdc1394 \
--enable-libdirac --enable-libfaac \
--enable-nonfree --enable-libfaad --enable-libgsm \
--enable-libmp3lame --enable-libopenjpeg \
--enable-libschroedinger --enable-libspeex --enable-libtheora \
--enable-libvorbis --enable-libx264 \
--enable-libxvid --enable-vdpau --enable-x11grab \
--enable-avfilter --enable-avfilter-lavf \
--enable-postproc --enable-swscale --enable-pthreads \
--disable-static --enable-shared \
--enable-gpl --disable-debug --disable-stripping \
--shlibdir=/usr/lib --cpu=i586 --disable-mmx2 \
--disable-sse --disable-ssse3 --disable-yasm
```

Once installed, here are some common use cases:

# get info about input file
```
% ffmpeg -i InputFile.mpg
```

# convert .wav to .mp3 with at 128 kbps
```
% ffmpeg -i Input.wav -ab 128 Output.mp3
```

# remove unneeded audio streams from file and change .avi container to mp4 container
```
% ffmpeg -i InputFile.avi -map 0:0 -map 2:1 -b 1200 OutputFile.mp4
    # -map 0:0 maps the video input stream 0 to output stream 0
    # -map 2:1 maps the audio input stream 2 to output stream 1
```

# Extract audio track to mp3
```
% ffmpeg -i MyMultitracAudio.m4a -map 0:0 MySingletracAudio.mp3
```

# Remove unneeded audio tracks from .m4a
```
% ffmpeg -i MyMultiTracAudio.m4a -map 0:0 -acodec copy \
MySingleTracAudio.m4a
```

# Convert WMV to MP4
```
% ffmpeg -i MyDeprecatedWMVSupportFile.wmv OutputFile.mp4
```

Good Encoding parameters for H.264 video and aac audio:
We have found good success with the following x264 presets that can be saved in the file /usr/share/ffmpeg/libx264-roku.ffpreset:

```
% cat /usr/share/ffmpeg/libx264-roku.ffpreset
coder=1
flags=+loop
cmp=+chroma
partitions=+parti8x8+parti4x4+partp8x8+partb8x8
me_method=umh
subq=8
me_range=16
g=250
keyint_min=25
sc_threshold=40
i_qfactor=0.71
b_strategy=2
qcomp=0.6
qmin=10
qmax=51
qdiff=4
bf=4
refs=4
directpred=3
trellis=1
flags2=+wpred+mixed_refs+dct8x8+fastpskip
```

With the above preset file, the **command for transcoding your files with high quality one pass constrant rate factor:**

```
% ffmpeg -i <InputFile> -acodec libfaac –ab 128 -vcodec libx264 \
 -vpre roku -crf 15 -threads 0 ~/outputFile.mp4
```

3.2 Audio / Video Interleave

The Roku Box does not support playing content that does not interleave the audio and the video. Content that does not sufficiently interleave the audio and video will show symptoms of constant rebuffers after a certain amount of time into the stream (typically ~15-60 min into the stream).

The maximum amount of contiguous video that may appear without audio must not exceed 5 megabytes and ideally should not exceed 1.5 megabytes. The maximum amount of contiguous audio that may appear without video must not exceed 512 kilobytes. In order to meet these constraints, a good rule of thumb is that the audio / video interleave should be less than 5 seconds.

The Adobe Media Encoder is one that by default does not interleave the audio and video. If you are using the Adobe Media Encoder, you could set the “stream compatibility” setting to “ipod” in the multiplexer settings rather than the default “standard”. This will cause the audio and video to be interleaved.

3.3 BIF File Creation

Please refer to the You'll likely want to generate two .bif archives for each piece of content, one for SD and one for HD. Which archive is used depends not on the resolution of the content, but on the resolution of the user's UI. That's why it's important to generate HD .bif archives even if the content is SD. However, if there is no HD .bif available, the player will fallback to using the SD .bif.

So here's one way to generate them:
```
% mkdir abc-sd abc-hd
% ffmpeg -i abc.mp4 -r .1 -s 240x180 abc-sd/%08d.jpg
% ffmpeg -i abc.mp4 -r .1 -s 320x240 abc-hd/%08d.jpg
% biftool -t 10000 abc-sd
% biftool -t 10000 abc-hd
```
This will result in two new files: abc-sd.bif and abc-hd.bif

There are two caveats here:
1) ffmpeg generates the .jpg files starting with index 1. This means that all the timestamps will be off by 10 seconds. To fix, use:

```
% j=00000000.jpg; for i in *; do mv ${i} ${j}; j=${i}; done;
```

2) The SD frames should have a width of 240 and the HD frames should have a width of 320. Their height should be specified to coincide with their aspect ratio. The commands above assume a 4x3 aspect ratio. Unfortunately, ffmpeg doesn't let you specify only a width, keeping the original aspect ratio. If your source file is not 4:3, you should calculate the height used in the commands above using the width and the aspect ratio.

4.0 HLS HTTP Live Streaming

HLS is the Roku recommended way of streaming video content. The Roku box is an advanced HLS player that will dynamically, proactively pick the appropriate bitrate segment to play based on the bandwidth available to the player. Many HLS players today do not dynamically adapt as the Roku Streaming Player does. HLS is the only way to deliver live content to the Roku Streaming Player, and will soon be the only way to do ad insertion without buffering events on the Roku Streaming Player.

The adaptive playback of HLS does put additional burdens on the encoding and distribution network used for streaming content. When encoding, many of the encoding parameters for each stream need to match. The framerate for all streams should match. The resolutions for HD streams should match and the resolution for all SD streams should match. At discontinuity points, marked by EXT-X-DISCONTINUITY tags in the m3u8 file, the resolution can now change.

Note that currently, BIF files cannot be used with HLS. This is an area we plan to address in a future release.

Please refer to the Pantos HTTP Live Streaming IETF specification for more detailed information about HLS: http://tools.ietf.org/html/draft-pantos-http-live-streaming-03. Roku is currently compatible with version 3 of this specification.

FFMpeg can also be a useful tool for transcoding content into a format compatible with HLS. HLS requires H.264 video and AAC-LC audio in MPEG Transport Stream segments. **The first step to transcoding can be accomplished with the command:**

```
% ffmpeg -i <inputFile> -f mpegts -acodec libfaac -vcodec libx264 \
 -vpre roku -crf 15 -threads 0 ~/OneTransportStreamFile3200.ts
```

Next, you need to “segment” the large transport stream file into multiple transport stream files each of the same “segment size” length. A typical segment size is 10 seconds. It is important that the segmenter you use will segment the transport stream on “I” frame boundaries. There are some open source segmenters out there, but we are not currently aware of any that segment properly on “I” frame boundaries. A segmenter we’ve had good success with is Apple’s. It comes by default in Mac OS X 10.6 (Snow Leopard) in the /usr/bin directory. It can be run like so:

```
% mediafilesegmenter –b http://yourhost.com/segmentsDirectory -t 10 \
 -l -i playlist3200.m3u8 –f ~/OneTransportStreamFile3200.ts
```

The command above will break OneTransportStreamFile.ts into multiple segments each 10 seconds in duration and save links to those segments in the playlist3200.m3u8 file. The base URL used to point to those segments in the playlist3200.m3u8 file is http://yourhost.com/segmentsDirectory. The mediafilesegmenter command can even do the transcoding from .mp4 and quicktime files, eliminating the previous ffmpeg command in some instances. You will probably run this command a couple more times for the other bitrate streams. Be careful to use the same segment duration for each playlist.

```
% mediafilesegmenter –b http://yourhost.com/segmentsDirectory -t 10 \
 -l -i \ playlist2400.m3u8  –f ~/OneTransportStreamFile2400.ts

% mediafilesegmenter –b http://yourhost.com/segmentsDirectory -t 10 \
 -l -i \ playlist1200.m3u8 –f ~/OneTransportStreamFile1200.ts
```

Finally, you will want to create a variant playlist that points to the three different bitrate playlists. Apple has a good tool to aid the creation of this file as well:

```
% variantplaylistcreator –o variantPlaylist.m3u8 \
http://yourhost.com/segmentsDirectory/playlist3200.m3u8 \
OneTransportStreamFile3200.plist \
http://yourhost.com/segmentsDirectory/playlist2400.m3u8 \
OneTransportStreamFile2400.plist \
http://yourhost.com/segmentsDirectory/playlist1200.m3u8 \
OneTransportStreamFile1200.plist
```

4.1 Encryption

Section 6.2.3 of the HLS specification gives details about the AES 128 bit encryption that HLS can use to protect your content. The Apple tools above can also be used to encrypt your segments. Here are the same commands adding encryption:

```
% mediafilesegmenter –b http://yourhost.com/segmentsDirectory -t 10 \
-l -i playlist3200.m3u8 –f ~/OneTransportStreamFile3200.ts –K \
https://yourhost.com/keyURL -J random
```

The –K option lets you specify the URL to use for the encryption key. You will want to protect access to the key, and a good authentication scheme is necessary. Please see the security write-up about client certificates and mutual authentication SSL in the Component Reference roUrlTransfer writeup. The –J command specifies a random IV (Initialization Value) for each segment’s encryption. The IV value for each segment is included in the .m3u8 file. The same –K and –J options can be added to the other two mediafilesegmenter commands. The variantplaylistcreater command remains the same.

4.2 Live Content

HLS is the only way to deliver live (unrecorded) video content to the Roku Streaming Player. There are a couple of things to be aware of when delivering live content. First, you must choose a Window Size that allows for variance in upload times to your CDN (Content Delivery Network). The minimum Window Size the HLS specification requires is three. However, we’ve found that a better number in practice is eight. The Window Size is the number of segments the .m3u8 file points to at a given time. A larger window size adds more delay to the overall delivery of the live stream. A smaller Window Size is less tolerant to jitter in the uploading of segments.

Second, you must be sure that your encoder and network have the capacity to deliver all your streams for the live event in a timely manner. We've created the script below that you can use to monitor all variant playlists to your live stream. You will want to check that each playlist is delivering the same segments on each round. It is OK if they are off by one as it takes a couple seconds for the script to get the .m3u8 files and headers for each playlist and corresponding segments. As long as the playlists seem to stay in sync and the content-length for each segment seems to be the correct size, your live content should play on the Roku without a hitch.

If you notice that the segment numbers drift so that one of the bitrates gets behind the others, you probably do not have the encoding capacity or upload bandwidth to your CDN to deliver all the streams at the bitrates you have chosen. At this point you will need to increase your capacity or dial down the number of variant playlists and decrease the bitrates of the streams you are delivering.

In the script below, you will modify the URLs to point to each .m3u8 in your variant playlist file:

```
#!/bin/bash

while true
do
echo "***************************************************************"
date
echo "***************************************************************"
wget -O - http://yourhost.com/playlist3200.m3u8 2>/dev/null | \
grep http | grep -v EXT-X-KEY > /tmp/playlist3200.txt
wget -O - http://yourhost.com/playlist2400.m3u8 2>/dev/null | \
grep http | grep -v EXT-X-KEY > /tmp/playlist2400.txt
wget -O - http://yourhost.com/playlist1200.m3u8 2>/dev/null | \
grep http | grep -v EXT-X-KEY > /tmp/playlist1200.txt

echo "**********"
cat /tmp/playlist3200.txt | wc -l
head -1 /tmp/playlist3200.txt
tail -1 /tmp/playlist3200.txt
for url in `cat /tmp/playlist3200.txt` ; do
    curl -I $url 2>/dev/null | grep 'Content-Length'
done
echo "**********"
cat /tmp/playlist2400.txt | wc -l
head -1 /tmp/playlist2400.txt
tail -1 /tmp/playlist2400.txt
for url in `cat /tmp/playlist2400.txt` ; do
    curl -I $url 2>/dev/null | grep 'Content-Length'
done
echo "**********"
cat /tmp/playlist1200.txt | wc -l
head -1 /tmp/playlist1200.txt
tail -1 /tmp/playlist1200.txt
for url in `cat /tmp/playlist1200.txt` ; do
    curl -I $url 2>/dev/null | grep 'Content-Length'
done
echo "**********"
```

```
sleep 10
done
```

4.3 Ad Insertion

Currently, our HLS playlists must be created from the same encode. All segments in an HLS Playlist must have monotonically increasing Presentation Time Stamp (PTS) values. This will not be true if some of the segments were encoded separately. A change in resolution at the discontinuity points is also currently a problem for the Roku Streaming Player.

In the very near future, we plan to eliminate these restrictions so that segments from different encodes can be intermixed in the same HLS playlist at discontinuity points. This will enable ads to become playable without issue or rebuffering in between the segments.

4.4 Inlet Encoder

We have found the Inlet Encoder to be very popular in the field. In this section and Appendix A, we will provide settings that we have found to be effective in our testing. Although this is still the cutting edge and there are often scalability problems to be overcome (please see Section 4.2 Live Content), we now have a little bit of experience to share with other developers.

As of this writing, the current version of Inlet software is 5.2. We ran into issues with all previous versions, so we recommend you upgrade.

It is important to select an Inlet encoder that has the capacity to encode all your streams simultaneously. Those broadcasters delivering HD sporting events are generally selecting the high end Inlet Spinnaker HD-X 8100: http://www.inlethd.com/?q=products/spinnaker/s8100

Please see Appendix A for an example roku.settings file that can be edited (you must at least change the _name, and model number). The sample configuration takes an HD 1920x1080 input and generates four bitrate streams. The streams referenced in the variant playlist are:

#1 800 kbps	SD
#2 1.4 mbps	SD
#3 2.2 mbps	SD
#4 2.8 mbps	HD 720p

4.5 Sample HD Content as input into your Inlet

When testing, not everyone has a Red camera (http://www.red.com/) readily available to feed HD content directly to your Inlet encoder. A more practical means of testing is to take a previously encoded stream (like HDMI out from your cable box) and pump it back to the Inlet. You will need a converter to do this, as the HD input for the Inlet is HD-SDI. Here are some HDMI to HD-SDI converters that may be of use to you in your encoding lab:
http://www.aja.com/products/converters/converters-hd-ha5.php
http://www.blackmagic-design.com/products/miniconverters/

4.6 Wowza Server

We've found the Wowza Server (http://www.wowzamedia.com) to be another very popular, budget minded choice in the HLS field. It's also common with content providers that need to produce their content in multiple formats like Flash and HLS. Here's a quick start guide to getting Wowza up and running with the Roku Streaming Player. We're just covering a quickstart here, but Wowza's forums have a wealth of information and that is a good place to search for solutions.

Earlier versions of the Wowza server had issues with not properly tagging audio as AAC-LC even when it was. Currently, Wowza exports a default bitrate of 64 kbps in the .m3u8 file. The best work around for this is to specify the actual bitrates for your content so that the buffering algorithms and stream selection algorithms on the Roku work properly. You can also specify a different default bandwidth for Wowza streams as discussed in this Wowza forum topic: http://www.wowzamedia.com/forums/showthread.php?t=9680. If you don't have access to the Wowza server, you could modify the Roku channel and set the content metadata parameter "minBandwidth" to 10. This would allow the Roku to play streams with bitrates higher than 10 kbps rather than the default, 250 kbps.

We've also found better results using the "unaligned-segments" switching strategy with vod Wowza. It's segments tend not to be aligned and it periodically sends a premature EOF on segment downloads. Other switching strategies may force a "skip" in time in the error handling of the premature segment EOF in the face of unaligned segments. This switching strategy does not work so well with Wowza live streams.

Wowza's name for HLS support in their server is "Cupertino streaming". They have a quick start guide on their site that they tend to keep up to date:
http://www.wowzamedia.com/quickstart_2_1_2.html

Using that guide for background information, you can quickly:

1. Download and install a free developer version of their server software: http://www.wowzamedia.com/store.html
2. The [install-dir] below will be:
   a. Linux: `/usr/local/WowzaMediaServer`
   b. Mac: `/Library/WowzaMediaServer`
   c. Windows: `C:\Program Files\Wowza Media Systems\Wowza Media Server 2.1.2`
3. `mkdir` **`[install-dir]/applications/vod`**
4. `mkdir` **`[install-dir]/content`**
5. `mkdir` **`[install-dir]/conf/vod`**
6. Create the file: **`[install-dir]/conf/vod/Application.xml`**

```
<Root>
        <Application>
                <Connections>
                        <AutoAccept>true</AutoAccept>
                        <AllowDomains></AllowDomains>
                </Connections>
                <Streams>
                <StreamType>default</StreamType>
                <StorageDir>/usr/local/WowzaMediaServer/content</StorageDir>
                <KeyDir>/usr/local/WowzaMediaServer/keys</KeyDir>
                <LiveStreamPacketizers></LiveStreamPacketizers>
                        <Properties>
                        </Properties>
                </Streams>
                <HTTPStreamers>cupertinostreaming</HTTPStreamers>
                <SharedObjects>
                        <StorageDir></StorageDir>
                </SharedObjects>
                <Client>
                        <IdleFrequency>-1</IdleFrequency>
                        <Access>
                        <StreamReadAccess>*</StreamReadAccess>
                        <StreamWriteAccess>*</StreamWriteAccess>
                          <StreamAudioSampleAccess></StreamAudioSampleAccess>
```

```
                    <StreamVideoSampleAccess></StreamVideoSampleAccess>
                    <SharedObjectReadAccess>*</SharedObjectReadAccess>
                    <SharedObjectWriteAccess>*</SharedObjectWriteAccess>
                </Access>
            </Client>
            <RTP>
                <!-- RTP/Authentication/[type]Methods defined in
                  Authentication.xml. Default setup includes; none, basic, digest -->
                <Authentication>
                        <PublishMethod>none</PublishMethod>
                        <PlayMethod>none</PlayMethod>
                </Authentication>
                <!-- RTP/AVSyncMethod. Valid values are: senderreport,
                      systemclock, rtptimecode -->
                <AVSyncMethod>senderreport</AVSyncMethod>
                <MaxRTCPWaitTime>12000</MaxRTCPWaitTime>

    <IncomingDatagramPortRanges>*</IncomingDatagramPortRanges>
            <!-- Properties defined here will override any properties defined  in
                  conf/RTP.xml for any depacketizers loaded by this application -->
                <Properties>
                </Properties>
            </RTP>
            <MediaCaster>
            <!-- Properties defined here will override any properties defined in
                  conf/MediaCasters.xml for any MediaCasters loaded by this
                  applications -->
                <Properties>
                </Properties>
            </MediaCaster>
            <MediaReader>
                <!-- Properties defined here will override any properties defined
                       in conf/MediaReaders.xml for any MediaReaders loaded by
                       this applications -->
                <Properties>
                </Properties>
            </MediaReader>
            <LiveStreamPacketizer>
                <!-- Properties defined here will override any properties defined
                      in conf/LiveStreamPacketizers.xml for any
                      LiveStreamPacketizers loaded by this applications -->
                <Properties>
                </Properties>
            </LiveStreamPacketizer>
            <HTTPStreamer>
                <!-- Properties defined here will override any properties defined
                      in conf/HTTPStreamers.xml for any HTTPStreamer loaded
                      by this applications -->
                <Properties>
                </Properties>
            </HTTPStreamer>
            <Repeater>
                <OriginURL></OriginURL>
                <QueryString><![CDATA[]]></QueryString>
            </Repeater>
            <Modules>
```

```
                    <Module>
                        <Name>base</Name>
                        <Description>Base</Description>
                        <Class>com.wowza.wms.module.ModuleCore</Class>
                    </Module>
                    <Module>
                        <Name>properties</Name>
                        <Description>Properties</Description>
                        <Class>com.wowza.wms.module.ModuleProperties</Class>
                    </Module>
                    <Module>
                     <Name>logging</Name>
                     <Description>Client Logging</Description>
                     <Class>com.wowza.wms.module.ModuleClientLogging</Class>
                    </Module>
                    <Module>
                      <Name>flvplayback</Name>
                      <Description>FLVPlayback</Description>
                      <Class>com.wowza.wms.module.ModuleFLVPlayback</Class>
                    </Module>
               </Modules>
               <!-- Properties defined here will be added to the
                    IApplication.getProperties() and IApplicationInstance.getProperties()
                    collections -->
               <Properties>
            <Property>
              <Name>smoothStreamingMediaMajorVersion</Name>
              <Value>2</Value>
              <Type>Integer</Type>
            </Property>
               </Properties>
          </Application>
     </Root>
```

7. You can now place your VOD content in **[install-dir]/content.** As an example we put big_buck_bunny_720p_h264.mov in the directory.
8. Modify the simplevideoplayer sample Roku app with a URL of the form: **http://192.168.1.23:1935/vod/mp4:big_buck_bunny_1080p_h264.mov/playlist.m3u8**

Appendix A: Inlet Spinnaker Sample Settings

In order to load the settings file below, save it in the file roku.settings and edit any parameters you feel need changing. You should at least change the _name, model, and version parameters to match your Inlet. Then, you can load it on the Inlet.

```
_name=roku-test
model=Spinnaker 7100
version=5.2.0.14436
max_vc1_encodes=4
max_avc_encodes=4
max_audio_encodes=4
max_h263_encodes=4
audio_source_type=2
```

```
audio_level_N=10,10,0,0,0,0,0,0
audio_upper_group=false
video_source_type=1
input_height=1080
input_width=1920
top_field_first=false
input_rateindex=4
input_scantype=0
IP_Address1=000.000.000.000
IP_Port1=0
IP_SourceAddress1=000.000.000.000
IP_Address2=000.000.000.000
IP_Port2=0
IP_SourceAddress2=000.000.000.000
IP_VideoPID=0
IP_AudioPID1=0
IP_DPIPID1=0
AdMarkerSlateFilename=
AdMarkerEnableFileSlate=false
AdMarkerEnableBlackSlate=false
crop_left=0
crop_right=1920
crop_top=0
crop_bottom=1080
output_height=240
output_width=320
video_encoding_profile=1
video_encoding_mode=0
target_bitrate=700
max_keyframe_distance=2000
video_max_bitrate=700
interlaced_output=false
buffer_delay=2000
quality_factor=80
complexity=0
deinterlacing=false
inverse_telecine=false
inlet_prefilter=0
black_and_white=false
cc_settings=0,0,0,0,0,0
open_captions=false
sami_enabled=false
ccscript_enabled=false
teletext_enabled=false
teletext_mode=0
wmv_output_file=
asf_network_port=8080
asf_network_max_clients=50
asf_packet_size=8000
asf_pub_point=
asf_push_server=
asf_push_template=
asf_auto_remove_pub_point=false
ts_output_file=
enable_tsfile=false
enable_multicast=false
```

```
multicast_address=224.0.0.1
multicast_packets=7
multicast_port=25000
multicast_ttl=8
padding_on=true
pmt_interval=400
seq_interval=33
ts_audio_stream_pesid=301
ts_program_id=256
ts_program_number=1
ts_video_stream_id=256
ts_video_stream_pesid=300
multicast_pip_address=224.0.0.1
multicast_pip_port=25010
ts_pip_program_id=257
ts_pip_program_number=1
ts_pip_video_stream_id=257
ts_pip_video_stream_pesid=300
vc1_output_enabled=false
flv_output_enabled=false
avc_output_enabled=true
vc1_mbr_separate=true
vc1_mbr_enabled_n=0,0,0,0
crop_left_n=0,0,0,0
crop_right_n=1920,1920,1920,1920
crop_top_n=0,0,0,0
crop_bottom_n=1080,1080,1080,1080
output_height_n=240,240,240,240
output_width_n=320,320,320,320
vc1_AspectRatioOverrideEnabled_n=0,0,0,0
vc1_AspectRatioOverride_n=65537,65537,65537,65537
video_encoding_profile_n=1,1,1,1
video_encoding_mode_n=0,0,0,0
target_bitrate_n=700,700,700,700
max_keyframe_distance_n=2000,2000,2000,2000
video_max_bitrate_n=700,700,700,700
interlaced_output_n=0,0,0,0
buffer_delay_n=2000,2000,2000,2000
quality_factor_n=80,80,80,80
complexity_n=0,0,0,0
deinterlacing_n=0,0,0,0
inverse_telecine_n=0,0,0,0
inlet_prefilter_n=0,0,0,0
black_and_white_n=0,0,0,0
deinterlacing_mode_n=0,0,0,0
resize_mode_n=0,0,0,0
resize_alg_n=0,0,0,0
frame_rate_divider_n=1,1,1,1
vc1_LoopFilter_n=-1,-1,-1,-1
vc1_NoiseFilter_n=-1,-1,-1,-1
vc1_MedianFilter_n=-1,-1,-1,-1
vc1_NoiseEdgeRemoval_n=-1,-1,-1,-1
vc1_OverlapFilter_n=-1,-1,-1,-1
vc1_NumBFrames_n=0,0,0,0
vc1_VariableGOP_n=-1,-1,-1,-1
vc1_ClosedGOP_n=-1,-1,-1,-1
```

```
vc1_LetterBoxPresent_n=-1,-1,-1,-1
vc1_KeyPopReduction_n=-1,-1,-1,-1
vc1_Lookahead_n=-1,-1,-1,-1
vc1_LookaheadRC_n=0,0,0,0
vc1_NoDropFrame_n=-1,-1,-1,-1
vc1_AdaptiveQuant_n=-1,-1,-1,-1
vc1_DquantOption_n=-1,-1,-1,-1
vc1_PFrameDQuantStrength_n=-1,-1,-1,-1
vc1_BFrameDQuantStrength_n=-1,-1,-1,-1
vc1_ForceBFrameDeltaQP_n=-1,-1,-1,-1
vc1_MBModeCost_n=-1,-1,-1,-1
vc1_MotionMatch_n=-1,-1,-1,-1
vc1_MotionSearchLevel_n=-1,-1,-1,-1
vc1_MotionSearchRange_n=-1,-1,-1,-1
vc1_MVCodingMethod_n=-1,-1,-1,-1
vc1_MVCostMethod_n=-1,-1,-1,-1
vc1_VideoEncodeType_n=-1,-1,-1,-1
vc1_NumThreads_n=-1,-1,-1,-1
vc1_AffinityMask_n=-1,-1,-1,-1
cc_settings_n=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
audio_settingsEx_n=eng,1,192,2,48000,0,33,eng,0,0,0,0,0,0,eng,0,0,0,0,0,0,eng,1,192,2,48000,
0,33,eng,0,0,0,0,0,0,eng,0,0,0,0,0,0,eng,1,192,2,48000,0,33,eng,0,0,0,0,0,0,eng,0,0,0,0,0,0,eng,
1,192,2,48000,0,33,eng,0,0,0,0,0,0,eng,0,0,0,0,0,0
audio_id_n=track_1,track_2,track_3,track_1,track_2,track_3,track_1,track_2,track_3,track_1,trac
k_2,track_3
wmv_output_file_n=-,-,-,-
wmv_output_file_enabled_n=0,0,0,0
unique_wmv_output_file_n=0,0,0,0
asf_pull_enabled_n=1,0,0,0
asf_network_port_n=8080,8082,8084,8086
asf_network_max_clients_n=50,50,50,50
asf_packet_size_n=8000,8000,8000,8000
asf_push_enabled_n=0,0,0,0
asf_pub_point_n=-,-,-,-
asf_push_server_n=-,-,-,-
asf_pub_point_aux_n=-,-,-,-
asf_push_server_aux_n=-,-,-,-
asf_push_template_n=-,-,-,-
asf_auto_remove_pub_point_n=0,0,0,0
asf_script_enabled_n=0,0,0,0
asf_script_bitrate_n=5,5,5,5
asf_script_flags_n=0,0,0,0
asf_title_n=-,-,-,-
asf_author_n=-,-,-,-
asf_copyright_n=-,-,-,-
asf_description_n=-,-,-,-
vc1_smoothPushEnabled_n=0,0,0,0
vc1_smoothStreamManifestFilename_n=-,-,-,-
vc1_smoothStreamManifestFilenameAux_n=-,-,-,-
vc1_smoothCustomVideoParamXML_n=-,-,-,-
vc1_smoothMaxSizeOverrideEnabled_n=0,0,0,0
vc1_smoothMaxSizeOverrideWidth_n=0,0,0,0
vc1_smoothMaxSizeOverrideHeight_n=0,0,0,0
vc1_smoothFragmentDuration_n=2000,2000,2000,2000
vc1_smoothFragmentDelay_n=0,0,0,0
vc1_smoothAudioTracks_n=1,0,0,0
```

```
vc1_smoothMetadata_0=0,Metadata,video,DATA,-,0,1000
vc1_smoothMetadata_4=0,AdMarkers,video,SCTE35,-,1,1000
vc1_smoothMetadata_5=0,captions,video,CAPT,TEXT,1,1000
VC1_PushUserName_N=-,-,-,-
VC1_PushPassword_N=-,-,-,-
VC1_PushUserNameAux_N=-,-,-,-
VC1_PushPasswordAux_N=-,-,-,-
VC1_WatermarkEnable_N=0,0,0,0
VC1_WatermarkParams_N=-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0
PlayReadyEnabled_N=0,0,0,0
PlayReadyProvider_N=0,0,0,0
_PlayReadyKID_N=-,-,-,-
PlayReadySeed_N=-,-,-,-
PlayReadyContentKey_N=-,-,-,-
PlayReadyLAURL_N=-,-,-,-
PlayReadyLAUIURL_N=-,-,-,-
PlayReadyDomainServiceID_N=-,-,-,-
PlayReadyCustomXML_N=-,-,-,-
flash_crop_bottom=1080
flash_crop_left=0
flash_crop_right=1920
flash_crop_top=0
flash_output_height=240
flash_output_width=320
flash_target_bitrate=500
flash_sharpness=7
flash_max_keyframe_distance=120
flash_video_framerate_divider=1
flash_quantizer_max=45
flash_keyframe_data_target=112
flash_auto_keyframe=true
flash_auto_keyframe_threshold=70
flash_minimum_keyframe_distance=8
flash_allow_drop_frames=false
flash_drop_frame_watermark=20
flash_realtime_mode_speed=7
flash_noise_sensitivity=0
flash_start_buffer_level=200
flash_optimal_buffer_level=200
flash_max_buffer_level=200
flash_undershoot_percent=95
flash_inverse_telecine=false
flash_deinterlacing=false
flash_deinterlacing_mode=0
flash_inlet_prefilter=0
flash_black_and_white=false
flash_resize_mode=0
flash_resize_alg=0
flash_audio_settings_ex=128,0,44100,0,0
flash_flv_enable_file=false
flash_flv_unique_file=false
flash_flv_output_file=
flash_flv_enable_stream=false
flash_asf_pub_point=
flash_asf_push_server=
flash_push_username=
flash_push_password=
flash_push_authenticate_mode=0
flash_WatermarkEnable_N=0
flash_WatermarkParams_N=-,0,0,0,0,0,0,0
vc1_mosaic_enabled=false
vc1_mosaic_settings=0,0,320,0,180,1,8080,0,320,640,0,180,2,8081,0,640,960,0,180,3,8082,0,9
60,1280,0,180,4,8083,0,0,320,180,360,5,8084,0,320,640,180,360,6,8085,0,640,960,180,360,7,8
086,0,960,1280,180,360,8,8087,0,0,320,360,540,9,8088,0,320,640,360,540,10,8089,0,640,960,3
60,540,11,8090,0,960,1280,360,540,12,8091,0,0,320,540,720,13,8092,0,320,640,540,720,14,80
93,0,640,960,540,720,15,8094,0,960,1280,540,720,16,8095
AVC_Enabled_N=1,1,1,1
AVC_CropLeft_N=0,0,0,0
AVC_CropRight_N=1920,1920,1920,1920
AVC_CropTop_N=0,0,0,0
AVC_CropBottom_N=1080,1080,1080,1080
AVC_VideoOutputWidth_N=1280,960,848,640
AVC_VideoOutputHeight_N=720,540,480,360
AVC_AspectRatioOverrideEnabled_N=0,0,0,0
AVC_AspectRatioOverride_N=65537,65537,65537,65537
AVC_AdaptiveStreaming_N=1,1,1,1
AVC_VideoEncodingProfile_N=1,1,0,0
AVC_VideoBitRate_N=2800,2200,1400,800
AVC_MaxKeyFrameDistance_N=60,60,30,30
AVC_VideoBufferSize_N=1000,1000,1000,1000
AVC_VideoFrameRateDivider_N=1,1,2,2
AVC_Deinterlacing_N=0,0,0,0
AVC_InverseTelecine_N=0,0,0,0
AVC_InletPreFilter_N=0,0,0,0
AVC_DeinterlacingMode_N=0,0,0,0
AVC_ResizeMode_N=0,0,0,0
AVC_ResizeAlg_N=4,4,4,4
AVC_WeightedPredictions_N=0,0,0,0
AVC_WriteAUDelimiters_N=1,1,1,1
AVC_NumBFrames_N=0,0,0,0
AVC_PReferenceFrameCount_N=2,2,1,1
AVC_WriteSeqEndCode_N=1,1,1,1
AVC_SceneChange_N=0,0,0,0
AVC_EntropyEncoding_N=1,0,0,0
AVC_MBAdaptiveFrameField_N=0,0,0,0
AVC_MESubpelMode_N=2,2,2,2
AVC_MESearchRange_N=127,63,63,63
AVC_VideoEncodingLevel_N=31,31,31,31
AVC_MinQuant_N=1,1,1,1
AVC_MaxQuant_N=51,51,51,51
AVC_PushEnabled_N=0,0,0,0
AVC_PubPoint_N=-,-,-,-
AVC_PushServer_N=-,-,-,-
AVC_PushAuthenticationMode_N=0,0,0,0
AVC_PushUserName_N=-,-,-,-
AVC_PushPassword_N=-,-,-,-
AVC_PushDelay_N=0,0,0,0
AVC_FlashOnFIInterval_N=0,0,0,0
AVC_PushEnabled2_N=0,0,0,0
AVC_PubPoint2_N=-,-,-,-
AVC_PushServer2_N=-,-,-,-
AVC_PushAuthenticationMode2_N=0,0,0,0
```

```
AVC_PushUserName2_N=-,-,-,-
AVC_PushPassword2_N=-,-,-,-
AVC_MP4FileName_N=-,-,-,-
AVC_EnableMP4File_N=0,0,0,0
AVC_UniqueMP4File_N=0,0,0,0
AVC_audio_settingsEx_n=2,14,44100,3,0,33,1,2,14,44100,3,0,33,1,2,14,44100,3,0,33,1,2,14,44
100,3,0,33,1,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0
,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,
33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,480
00,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,19,48000,0,0,33,0,0,
19,48000,0,0,33,0,0,19,48000,0,0,33,0
AVC_AudioLang_N=eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,eng,e
ng,eng,eng,eng,eng,eng,eng
AVC_AudioID_N=track_1,track_2,track_3,track_4,track_5,track_6,track_1,track_2,track_3,track_
4,track_5,track_6,track_1,track_2,track_3,track_4,track_5,track_6,track_1,track_2,track_3,track_
4,track_5,track_6
AVC_MulticastEnabled_N=0,0,0,0
AVC_MulticastIPAddress_N=224.0.0.1,224.0.0.2,224.0.1.3,224.0.0.4
AVC_MulticastPackets_N=7,7,7,7
AVC_MulticastPort_N=27000,27001,14000,27003
AVC_MulticastTTL_N=8,8,8,8
AVC_AudioPID_N=482,482,482,482
AVC_ProgramID_N=480,480,480,480
AVC_ProgramNumber_N=1,1,1,1
AVC_VideoPID_N=481,481,481,481
AVC_PreferredMulticastNIC_N=2,2,2,2
AVC_TSFileName_N=c:\test2.ts,-,-,-
AVC_TSFileEnabled_N=0,0,0,0
AVC_UniqueTSFile_N=0,0,0,0
AVC_TSSingleArchive_N=1,1,1,1
AVC_TSSegmentedArchive_N=0,0,0,0
AVC_TSSegmentedArchiveDuration_N=0,0,0,0
AVC_TSManualArchive_N=0,0,0,0
AVC_AdaptiveTSEnabled_N=0,0,0,0
AVC_AdaptiveTSDuration_N=10,10,10,10
AVC_AdaptiveTSMetadataRate_N=0,0,0,0
AVC_AppleEnable_N=1,1,1,1
AVC_AppleStreamName_N=wifistream,3Ghistream,3GLowstream,Edgestream
AVC_AppleInternalURL_N=http://iphone.company.com.edgesuite.net/,http://iphone.company.co
m.edgesuite.net/,http://iphone.company.com.edgesuite.net/,http://iphone.company.com.edgesuite
.net/
AVC_AppleExternalURL_N=http://iphone.company.com.edgesuite.net/,http://iphone.company.co
m.edgesuite.net/,http://iphone.company.com.edgesuite.net/,http://iphone.company.com.edgesuite
.net/
AVC_AppleSegDuration_N=10,10,10,10
AVC_AppleSegPerIndex_N=10,10,10,10
AVC_AppleSegPerFolder_N=200,200,200,200
AVC_AppleSegLag_N=1,1,1,1
AVC_AppleSegToKeep_N=20,20,20,20
AVC_AppleSegEnableAkamai_N=0,0,0,0
AVC_AppleSegAkamaiToken_N=-,-,-,-
AVC_AppleSegAkamaiKey_N=-,-,-,-
AVC_AppleSegAkamaiTTL_N=0,0,0,0
AVC_AppleSegGenerateMaster_N=1,1,1,1
AVC_AppleSegMasterName_N=ipad,ipad|iphone,ipad|iphone,ipad|iphone
AVC_AppleInternalURL2_N=-,-,-,-
```

```
AVC_AppleExternalURL2_N=-,-,-,-
AVC_AppleSegEncryption_N=0,0,0,0
AVC_AppleSegUseExistingKey_N=0,0,0,0
AVC_AppleSegKeyPeriod_N=0,0,0,0
AVC_AppleSegKeyFileName_N=-,-,-,-
AVC_AppleSegKeyInternalURL_N=-,-,-,-
AVC_AppleSegKeyExternalURL_N=-,-,-,-
AVC_AppleSegKeyInternalURL2_N=-,-,-,-
AVC_AppleSegKeyExternalURL2_N=-,-,-,-
AVC_AppleVODMode_N=0,0,0,0
AVC_AppleGenerateAudio_N=0,0,0,1
AVC_AppleGenerateAudioImage_N=0,0,0,10
AVC_AppleAudioImage_N=-,-,-,-
AVC_GenerateAudioTS_N=0,0,0,0
AVC_RTP_Enabled_N=0,0,0,0
AVC_RTP_Server_N=-,-,-,-
AVC_RTP_StreamName_N=-,-,-,-
AVC_RTP_PushPort_N=0,0,0,0
AVC_RTP_VideoPort_N=0,0,0,0
AVC_RTP_AudioPort_N=0,0,0,0
AVC_RTP_AuthenticationEnabled_N=0,0,0,0
AVC_RTP_UserName_N=-,-,-,-
AVC_RTP_Password_N=-,-,-,-
AVC_RTP_Enabled2_N=0,0,0,0
AVC_RTP_Server2_N=-,-,-,-
AVC_RTP_StreamName2_N=-,-,-,-
AVC_RTP_PushPort2_N=0,0,0,0
AVC_RTP_VideoPort2_N=0,0,0,0
AVC_RTP_AudioPort2_N=0,0,0,0
AVC_RTP_AuthenticationEnabled2_N=0,0,0,0
AVC_RTP_UserName2_N=-,-,-,-
AVC_RTP_Password2_N=-,-,-,-
AVC_cc_settings_n=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
AVC_sami_enabled=false
AVC_ccscript_enabled=false
AVC_teletext_enabled=false
AVC_teletext_mode=0
AVC_NumThreads_N=0,0,0,0
AVC_ColorDescription_N=1,1,1,1
AVC_ColorPrimaries_N=6,6,6,6
AVC_TransferCharacteristics_N=1,1,1,1
AVC_MatrixCoefficients_N=6,6,6,6
AVC_AdvancedParamSelection_N=0,0,0,0
AVC_WatermarkEnable_N=0,0,0,0
AVC_WatermarkParams_N=-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0,-,0,0,0,0,100,0,0
AVC_SmoothPushEnable=0,0,0,0
AVC_SmoothPubPoint=-,-,-,-
AVC_SmoothServer=-,-,-,-
AVC_SmoothStreamManifest=-,-,-,-
AVC_SmoothPubPointAux=-,-,-,-
AVC_SmoothServerAux=-,-,-,-
AVC_SmoothStreamManifestAux=-,-,-,-
AVC_SmoothCustomVideoXML=-,-,-,-
AVC_SmoothUserName=-,-,-,-
AVC_SmoothPassword=-,-,-,-
AVC_SmoothUserNameAux=-,-,-,-
```

```
AVC_SmoothPasswordAux=-,-,-,-
AVC_SmoothLowLatencyEnabled=false
AVC_SmoothLowLatencyVFPF=10
AVC_SmoothLowLatencyAFPF=15
AVC_SmoothMaxSizeOverrideEnabled_n=0,0,0,0
AVC_SmoothMaxSizeOverrideWidth_n=0,0,0,0
AVC_SmoothMaxSizeOverrideHeight_n=0,0,0,0
AVC_smoothFragmentDuration_n=2000,2000,2000,2000
AVC_smoothFragmentDelay_n=0,0,0,0
AVC_SmoothAudioTracks_n=1,0,0,0
AVC_PlayReadyEnabled_N=0,0,0,0
AVC_PlayReadyProvider_N=0,0,0,0
AVC_PlayReadyKID_N=-,-,-,-
AVC_PlayReadySeed_N=-,-,-,-
AVC_PlayReadyContentKey_N=-,-,-,-
AVC_PlayReadyLAURL_N=-,-,-,-
AVC_PlayReadyLAUIURL_N=-,-,-,-
AVC_PlayReadyDomainServiceID_N=-,-,-,-
AVC_PlayReadyCustomXML_N=-,-,-,-
AVC_SmoothMetadata_0=0,Metadata,video,DATA,-,0,1000
AVC_SmoothMetadata_4=0,AdMarkers,video,SCTE35,-,1,1000
AVC_SmoothMetadata_5=0,captions,video,CAPT,TEXT,1,1000
H263_Enabled_N=0,0,0,0
H263_CropLeft_N=0,0,0,0
H263_CropRight_N=1920,1920,1920,1920
H263_CropTop_N=0,0,0,0
H263_CropBottom_N=1080,1080,1080,1080
H263_VideoOutputWidth_N=480,480,480,480
H263_VideoOutputHeight_N=480,480,480,480
H263_AspectRatioOverrideEnabled_N=0,0,0,0
H263_AspectRatioOverride_N=65537,65537,65537,65537
H263_VideoEncodingProfile_N=10,10,10,10
H263_VideoEncodingLevel_N=0,0,0,0
H263_VideoBitRate_N=64,64,64,64
H263_MaxKeyFrameDistance_N=60,60,60,60
H263_VideoBufferSize_N=1000,1000,1000,1000
H263_VideoFrameRateDivider_N=0,0,0,0
H263_Deinterlacing_N=0,0,0,0
H263_InverseTelecine_N=0,0,0,0
H263_InletPreFilter_N=0,0,0,0
H263_DeinterlacingMode_N=0,0,0,0
H263_ResizeMode_N=0,0,0,0
H263_ResizeAlg_N=0,0,0,0
H263_NumBFrames_N=0,0,0,0
H263_BitRateMode_N=0,0,0,0
H263_AMR_Enabled_N=1,1,1,1
H263_AMR_BitRate_N=4,4,4,4
H263_AMR_SampleRate_N=8000,8000,8000,8000
H263_AMR_Offset_N=0,0,0,0
H263_AMR_MonoMode_N=1,0,0,0
H263_AMR_ChannelMap_N=1,33,33,33
H263_RTP_Enabled_N=0,0,0,0
H263_RTP_Server_N=-,-,-,-
H263_RTP_StreamName_N=-,-,-,-
H263_RTP_PushPort_N=0,0,0,0
H263_RTP_VideoPort_N=0,0,0,0
```

```
H263_RTP_AudioPort_N=0,0,0,0
H263_RTP_AuthenticationEnabled_N=0,0,0,0
H263_RTP_UserName_N=-,-,-,-
H263_RTP_Password_N=-,-,-,-
H263_RTP_Enabled2_N=0,0,0,0
H263_RTP_Server2_N=-,-,-,-
H263_RTP_StreamName2_N=-,-,-,-
H263_RTP_PushPort2_N=0,0,0,0
H263_RTP_VideoPort2_N=0,0,0,0
H263_RTP_AudioPort2_N=0,0,0,0
H263_RTP_AuthenticationEnabled2_N=0,0,0,0
H263_RTP_UserName2_N=-,-,-,-
H263_RTP_Password2_N=-,-,-,-
AO_audio_channel_mapping_N=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
AO_enabled_streams_N=0,0,0,0
AO_FlashOnFIInterval_N=0,0,0,0
AO_streaming_0=8,2,-,-,0,-,-,-,0,65535,-1,50,-
AO_streaming_1=8,2,-,-,0,-,-,-,0,65535,-1,50,-
AO_streaming_2=8,2,-,-,0,-,-,-,0,65535,-1,50,-
AO_streaming_3=8,2,-,-,0,-,-,-,0,65535,-1,50,-
AO_archiving_0=8,-,-,0
AO_archiving_1=8,-,-,0
AO_archiving_2=8,-,-,0
AO_archiving_3=8,-,-,0
AO_encoding_0=7,128,44100
AO_encoding_1=7,128,12000
AO_encoding_2=7,128,12000
AO_encoding_3=7,128,12000
AO_attributes_0=-,-,-,0
AO_attributes_1=-,-,-,0
AO_attributes_2=-,-,-,0
AO_attributes_3=-,-,-,0
```