Documentation:Streaming HowTo/Command Line Examples

From VideoLAN Wiki
< Documentation:Streaming HowTo

Examples for advanced use of VLC's stream output (transcoding, multiple streaming, etc...)

Contents

Transcoding

Transcode a stream to Ogg Vorbis with 2 channels at 128kbps and 44100Hz and save it as *foobar.ogg*:

```
% vlc -I dummy -vvv input_stream --sout
"#transcode{vcodec=none,acodec=vorb,ab=128,channels=2,samplerate=44100}:file{dst=foobar.ogg}"
```

Transcode the input stream and send it to a multicast IP address with the associated SAP announce:

```
% vlc -vvv input_stream --sout
'#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128,deinterlace}:
rtp{mux=ts,dst=239.255.12.42,sdp=sap,name="TestStream"}'
```

Display the input stream, transcode it and send it to a multicast IP address with the associated SAP announce:

```
% vlc -vvv input_stream --sout
#duplicate{dst=display,dst="transcode{vcodec=mp4v,acodec=mpga,vb=800,
ab=128,deinterlace}:rtp{mux=ts,dst=239.255.12.42,sdp=sap,name="TestStream"}"}'
```

Transcode the input stream, display the transcoded stream and send it to a multicast IP address with the associated SAP announce:

```
% vlc -vvv input_stream --sout
'#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128,deinterlace}:
duplicate{dst=display,dst=rtp{mux=ts,dst=239.255.12.42,sdp=sap,name="TestStream"}}'
```

To receive the input stream that is being multicasted above on a client:

```
vlc rtp://239.255.12.42
```

More complex transcoding example

Stream a SDI card to H.264 and AAC in TS on udp

```
% cvlc -vvv --live-caching 2000 decklink://
--decklink-audio-connection embedded --decklink-aspect-ratio 16:9 --decklink-mode hp50
--sout-x264-preset slow --sout-x264-tune film --sout-transcode-threads 8 --no-sout-x264-interlaced
--sout-x264-keyint 50 --sout-x264-lookahead 100 --sout-x264-vbv-maxrate 6000 --sout-x264-vbv-bufsize 6000
--sout '#transcode{vcodec=h264,vb=6000,acodec=mp4a,aenc=fdkaac,ab=256}:std{access=udp,mux=ts,dst=192.168.2.1:
```

Multiple streaming

Send a stream to a multicast IP address and a unicast IP address:

```
vlc -vvv input_stream
--sout '#duplicate{dst=rtp{mux=ts,dst=239.255.12.42,sdp=sap,name="TestStream"},dst=rtp{mux=ts,dst=192.168.1.2
```

Display the stream and send it to two unicast IP addresses:

```
vlc -vvv input_stream
--sout '#duplicate{dst=display,dst=rtp{mux=ts,dst=192.168.1.12},dst=rtp{mux=ts,dst=192.168.1.42}}'
```

Send parts of a multiple program input stream:

```
vlc -vvv multiple_program_input_stream
```

```
--sout'#duplicate{dst=rtp{mux=ts,dst=239.255.12.42},select="program=12345",dst=rtp{mux=ts,dst=239.255.12.43},
```

This command sends the program of the input stream which id is 12345 to 239.255.12.42 and all video programs with id between 1234 and 2345 to 239.255.12.43.

Transcoding and multiple streaming

Transcode the input stream, display the transcoded stream and send it to a multicast IP address with the associated SAP announce and an unicast IP address:

```
% vlc -vvv input_stream --sout
'#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128,deinterlace}:
duplicate{dst=display,dst=rtp{mux=ts,dst=239.255.12.42,sdp=sap,name="TestStream"},
dst=rtp{mux=ts,dst=192.168.1.2}}'
```

Display the input stream, transcode it and send it to two unicast IP addresses:

```
% vlc -vvv input_stream --sout  '#duplicate{dst=display,dst="transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128}
duplicate{dst=rtp{mux=ts,dst=192.168.1.2},dst=rtp{mux=ts,dst=192.168.1.12}"}'
```

Send the input stream to a multicast IP address and the transcoded stream to another multicast IP address with the associated SAP announces:

```
% vlc -vvv input_stream --sout
'#duplicate{dst=rtp{mux=ts,dst=239.255.1.2,sdp=sap,name="OriginalStream"},
dst="transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128}:
rtp{mux=ts,dst=239.255.1.3,sdp=sap,name="TranscodedStream"}"}'
```

More complex multi-transcoding example

Take a SDI input, and transcode it twice, once in HD, and one in SD and send both on udp.

```
% cvlc -vv --live-caching 2000
--decklink-audio-connection embedded --decklink-aspect-ratio 16:9 --decklink-mode hp50 decklink://
--sout-x264-preset fast --sout-x264-tune film --sout-transcode-threads 24 --no-sout-x264-interlaced
--sout-x264-keyint 50 --sout-x264-lookahead 100 --sout-x264-vbv-maxrate 4000 --sout-x264-vbv-bufsize 4000
--sout '#duplicate{dst="transcode{vcodec=h264,vb=6000,acodec=mp4a,aenc=fdkaac,ab=256}:std{access=udp,mux=ts,d
dst="transcode{height=576,vcodec=h264,vb=2000,acodec=mp4a,aenc=fdkaac,ab=128}:std{access=udp,mux=ts,dst=192.1
```

Take a SDI input, and restreaming it once in raw and transcoding it for the second

```
% cvlc -vv --live-caching 2000
--decklink-audio-connection embedded --decklink-aspect-ratio 16:9 --decklink-mode hp50 decklink://
--sout-x264-preset fast --sout-x264-tune film --sout-transcode-threads 24 --no-sout-x264-interlaced
--sout-x264-keyint 50 --sout-x264-lookahead 100 --sout-x264-vbv-maxrate 4000 --sout-x264-vbv-bufsize 4000
--sout '#duplicate{dst="transcode{vcodec=h264,vb=6000,acodec=mp4a,aenc=fdkaac,ab=256}:std{access=udp,mux=ts,d
```

```
dst="std{access=udp,mux=ts,dst=192.168.1.2:4014}"}'
```

HTTP streaming

Stream in HTTP:

- on the server, run:

```
% vlc -vvv input_stream --sout '#standard{access=http,mux=ogg,dst=server.example.org:8080}'
```

- on the client(s), run:

```
% vlc http://server.example.org:8080
```

Transcode and stream in HTTP:

```
% vlc -vvv input_stream --sout '#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128}:
standard{access=http,mux=ogg,dst=server.example.org:8080}'
```

Recording a live video stream:

```
% vlc http://example.com/live.asf --sout="#duplicate{dst=std{access=file,mux=asf,
dst='C:\test\test.asf'},dst=nodisplay}"
```

For example, if you want to stream an audio CD in Ogg/Vorbis over HTTP:

```
% vlc -vvv cdda:/dev/cdrom
--sout '#transcode{acodec=vorb,ab=128}:standard{access=http,mux=ogg,dst=server.example.org:8080}'
```

RTSP live streaming

Stream with RTSP and RTP:

- Run on the server:

```
vlc -vvv input_stream --sout '#rtp{dst=192.168.0.12,port=1234,sdp=rtsp://server.example.org:8080/test.sdp}'
```

- Run on the client(s):

```
vlc rtsp://server.example.org:8080/test.sdp
```

RTSP on-demand streaming

See Documentation:Streaming HowTo/VLM.

MMS / MMSH streaming to Windows Media Player

```
% vlc -vvv input_stream --sout '#transcode{vcodec=DIV3,vb=256,scale=1,acodec=mp3,ab=32,
channels=2}:std{access=mmsh,mux=asfh,dst=:8080}'
```

VLC media player can connect to this by using the following url: **mmsh://server_ip_address:8080**. Windows Media Player can connect to this by using the following url: **mms://server_ip_address:8080**.

Use the *es* module

Separate audio and video in two PS files:

```
% vlc -vvv input_stream --sout '#es{access=file,mux=ps,url_audio=audio-%c.%m,url_video=video-%c.%m}'
```

Extract the audio track of the input stream to a TS file:

```
% vlc -vvv input_stream --sout '#es{access_audio=file,mux_audio=ts,url_audio=audio-%c.%m}'
```

Stream in unicast the audio track on a port and the video track on another port (NOTE: This will not only work with VLC 0.8.6 or older - FIXME?):[Please check this]

- on the server side:

```
% vlc -vvv input_stream --sout '#es{access=rtp,mux=ts,url_audio=192.168.1.2:1212,
url_video=192.168.1.2:1213}'
```

- on the client side:
  - to receive the audio:

```
% vlc udp://@:1212
```

  - to receive the video:

```
% vlc udp://@:1213
```

Stream in multicast the video and dump the audio in a file:

```
% vlc -vvv input_stream --sout '#es{access-video=udp,mux-video=ts,dst-video=239.255.12.42,
access-audio=file,mux-audio=ps,dst-audio=audio-%c.%m}'
```

Note: You can also combine the *es* module with the other modules to set-up even more complex solution.

Keeping the stream open

```
% % vlc -vvv input_stream -sout-keep
-sout=#transcode{acodec=mp3}:duplicate{dst=display{delay=6000},
dst=gather:std{mux=mpeg1,dst=:8080/stream.mp3,access=http},select="novideo"}
```

The basic transcoding is an mp3 stream from the file you select (if it is a video file, then the video is ignored). It is streamed via http to localhost:8080/stream.mp3

The combination of :sout-keep and dst=gather:std mean that the stream is kept open and subsequent items are played through the same stream.

Using VLC as a reflector

Taking a udp input and resending it once raw via ipv6 multicast, and once in HLS

```
% cvlc -vvv udp://@:4013 --ttl 60
--sout '#duplicate{dst=std{access=http,mux=ts,dst=[::]:3013}",
dst=std{access=udp,mux=ts,dst=ffe2::1]:2013},
dst=std{access=livehttp{seglen=5,delsegs=true,numsegs=5,index=/path/to/stream.m3u8,
index-url=http://example.org/stream-########.ts},mux=ts{use-key-frames},dst=/path/to/stream-########.ts}}}
```