1 Compilation:

g++: run make in ./linux/

2 Usage Information:

Binaries in ./linux/bin/

2.1 For non-segmented files:

```
Usage: ValidateMP4.exe [-filetype <type>] [-printtype <options>] [-checklevel <level>] [-infofile <Segment
Info File>] [-leafinfo <Leaf Info File>] [-segal] [-ssegal] [-startwithsap TYPE] [-level] [-bss] [-isolive]
[-isoondemand] [-isoma
 [-samplenumber <number>] [-verbose <options>] [-offsetinfo <Offset Info File>] [-logconsole ] [-help]
inputfile
    -a[tompath]      <atompath> - limit certain operations to <atompath> (e.g. moov-1:trak-2)
                     this effects -checklevel and -printtype (default is everything)
    -p[rinttype]     <options> - controls output (combine options with +)
                     atompath - output the atompath for each atom
                     atom - output the contents of each atom
                     fulltable - output those long tables (e.g. samplesize tables)
                     sample - output the samples as well
                                 (depending on the track type, this is the same as sampleraw)
                     sampleraw - output the samples in raw form
                     hintpayload - output payload for hint tracks
    -c[hecklevel]    <level> - increase the amount of checking performed
                     1: check the moov container (default -atompath is ignored)
                     2: check the samples
                     3: check the payload of hint track samples
    -infofile        <Segment Info File> - Offset file generated by assembler
    -leafinfo         <Leaf Info File> - Information file generated by this software (named leafinfo.txt)
for another representation, provided to run for cross-checks of alignment
    -segal  -         Check Segment alignment based on <Leaf Info File>
    -ssegal -         Check Subegment alignment based on <Leaf Info File>
    -bandwidth        For checking @bandwidth/@minBufferTime
    -minbuffertime    For checking @bandwidth/@minBufferTime
    -sbw              Suggest a good @bandwidth if the one provided is non-conforming
    -isolive          Make checks specific for media segments conforming to ISO Base media file format live
profile
    -isoondemand      Make checks specific for media segments conforming to ISO Base media file format On
Demand profile
    -isomain          Make checks specific for media segments conforming to ISO Base media file format main
profile
    -dynamic          MPD type=dynamic
    -startwithsap     Check for a specific SAP type as announced in the MPD
    -level            SubRepresentation@level checks
    -bss              Make checks specific for bitstream switching
    -dash264base      Make checks specific for DASH264 Base IOP
    -dash264enc       Make checks specific for encrypted DASH264 content
    -indexrange       Byte range where sidx is expected
    -width            Expected width of the video track
    -height           Expected height of the video track
    -s[amplenumber]   <number> - limit sample checking or printing operations to sample <number>
                      most effective in combination with -atompath (default is all samples)
    -offsetinfo       <Offset Info File> - Partial file optimization information file: if the file has
several byte ranges removed, this file provides the information as offset-bytes removed pairs
    -logconsole       Redirect stdout and stderr to stdout.txt and stderr.txt, respectively

    -h[elp] - print this usage message
```

Output and errors (if any) will be printed on the console.

Description of `-infofile` and `-leafinfo` is provided in the following sections.

2.2 For segmented files:

Segmented files must first be re-assembled. A script "Assemble" is provided with the following usage.

```
Assemble [1/0] (initialization segment) [segment 1] [segment 2]... [last
segment]
1: first file is an initialization segment, 0 otherwise.
```

Typical usage would look like:

```
Assemble 1 [initialization segment] [segment 1] [segment 2]... [last segment]
```

or

```
Assemble 0 [segment 1] [segment 2]... [last segment]
```

depending whether or not an initialization segment is provided. The entries in square brackets are the names of the respective segments. To ease this, place all the segments in the same folder as the script itself. This script generates an assembled file "tempMerged.mp4" and a corresponding segment information file "segmentSizeInfoFile.txt". The former file will be the inputfile and the latter will be used in conjunction with `-infofile` for segment validation.

Tip to make life easier: The above construct works reasonably well except when you start having to merge hundreds of segments for a representation. This can be automated easily e.g. by sorting the directory numerically and formatting the output as in the command:

```
ls <optional: representation identifier> | sort -V | tr '\n' ' '
```

This will make sure that, e.g. Segment 10 is listed *after* Segment 2. One can copy the relevant section of the output of the above command and use it to construct the command syntax for "Assemble" script above. Adding an optional representation identifier (e.g. *5000000* to select a 5 Mbps representation, if applicable) helps short listing further.

2.3 For segment/subsegment Alignment checks:

ValidateMP4.exe creates a leaf subsegment information file "leafinfo.txt". If segment or subsegment alignment of a representation B is to be cross checked with representation A:

1. Run `ValidateMP4.exe representation A`

2. Run `ValidateMP4.exe representation B -leafinfo leafinfo.txt -segal` or `ValidateMP4.exe representation B -leafinfo leafinfo.txt -ssegal` (for checking segment or subsegment alignment, respectively)

3 Features

Refer to features.doc file.