Watermark Robustness Test Process

Contents

1 Introduction

This document describes two tests: the Robustness Test (Sections 2 and 3) and the Complexity Test (Section 4). Each proponent shall conduct the Robustness Test and the Complexity Test themselves (i.e. self-test) using their own watermark technology, i.e. their own watermark embedder and watermark detector and the specified common methodology and content.

The function of the technology being tested is to identify the boundaries of watermarked segments. Segments are marked with the same MediaID (i.e. Ad-ID or EIDR), such that boundaries are identified by a change in MediaID.

The Robustness Test assesses the performance of watermarking technology with respect to the following evaluation criteria:

1.1.1 Payload

1.1.2 Survivability across Platforms

1.1.4 Granularity

1.1.5 Retrievability

1.1.6 ID Replacement

The Complexity Test assesses the performance of watermarking technology with respect to the following evaluation criteria:

1.1.13 Performance Impact

In the Robustness Test, the content presented to the watermark detector is intended to approximate broadcast programs and viewer behavior, i.e. the viewer "channel surfing" with channel changes, content changes within the same channel and media timeline changes caused by time-shifting (i.e., skip forward/back) as well as signal processing imposed by the broadcast chain (e.g. media encoding and decoding, etc.).

Proponents shall test their systems with proponent-selected parameters (e.g., payload size, payload frequency, error-correction mechanisms, etc.), provided that the parameters used shall satisfy the requirements contained in [1]. It is expected that the system could be able to be used "in the real world" with the parameter settings tested.

All testing must be performed using the identical configuration of the proponent watermark embedding and detecting systems (i.e. the configuration may not be tailored to the individual test items). The test processing, intermediate and final test files, detector log output files, "raw" detector output (if available), and data analysis procedures must be documented and preserved to enable a third-party to replicate and validate all reported results at a some future date.

2 Overview of Robustness Test Process

The following figures illustrate the processing steps and the intermediate and final results in the Robustness Test. The figures make use of terms defined in Section 2.1.

Figure 1, below, shows the process required to produce a Test Set. The Original Content consists of a number of content types, as shown in Annex A, Table 3.

The Original Content is marked by the proponent watermark embedder to produce Marked Content. The Marked Content and the Original Content is subject to a specific Processing Chain (denoted as Processing Chain N or abbreviated as PC-N). This might be audio encoding and decoding at a specified bit rate. The result is Processed Marked Content and Processed Original Content. The Test Set construction process takes Segments from the Processed Marked Content and Processed Original Content and produces a specific Test Set, denoted as TS-N where N is the index of the Processing Chain used. The Test Set construction process also produces a Reference Log, which is the output log file for a "perfect" detector running on TS-N.

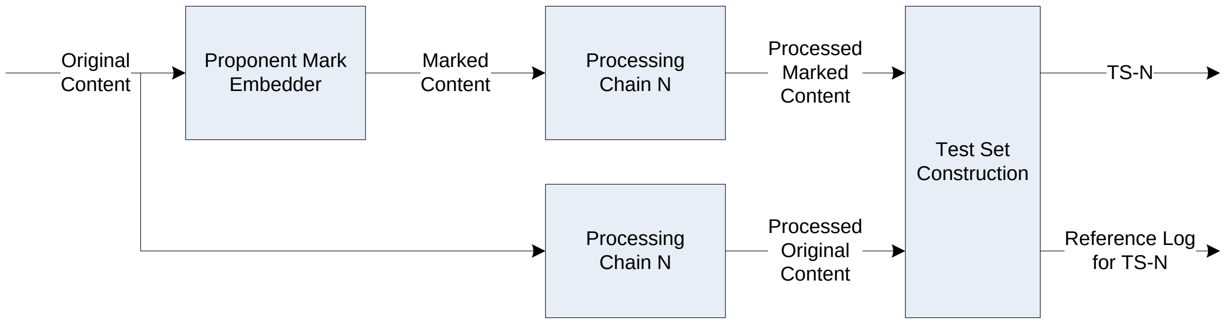


Figure 1 – Block Diagram of Marking, Processing and Test Set Construction Process

Figure 2, below, shows the detailed process used to construct the Test Set. The content types or categories (see Annex A) are shown in different colors. The Test Set is constructed by taking Segments with randomly chosen start and end times from the content and concatenating these to make the Test Set. Segments can be selected from any content type and from either the Processed Marked Content or the Processed Original Content. Segments may be followed by a "gap" of "zero audio" that simulates the time required for a set top box to acquire the new channel signal. A detailed test set construction process is specified in the scripts documented in Annex D.

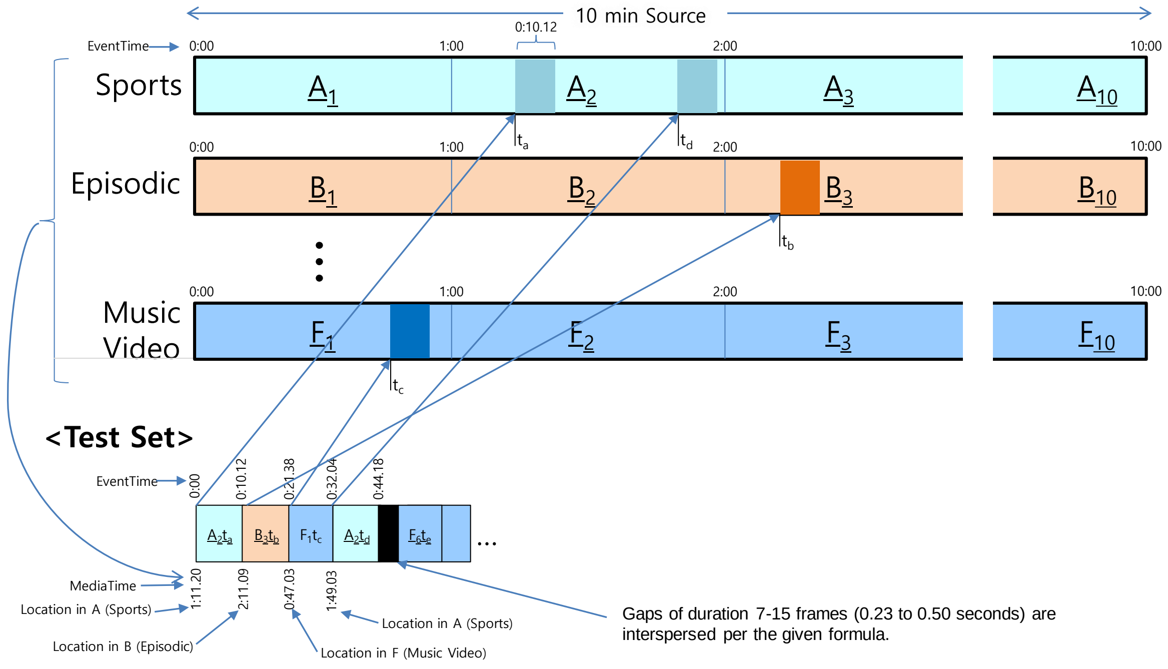


Figure 2 – Detail of Test Set Construction Process (note that annotation details do not necessarily apply to this test procedure)

Figure 3, below, shows the process required to produce a Performance Report on a proponents mark technology for a given Test Set. The proponent watermark detector is run on a specific Test Set (e.g. TS-N) to produce a Detector Log for TS-N. This Detector Log is compared to the Reference Log for TS-N and differences in detector output are compiled into a report on Mark Performance.

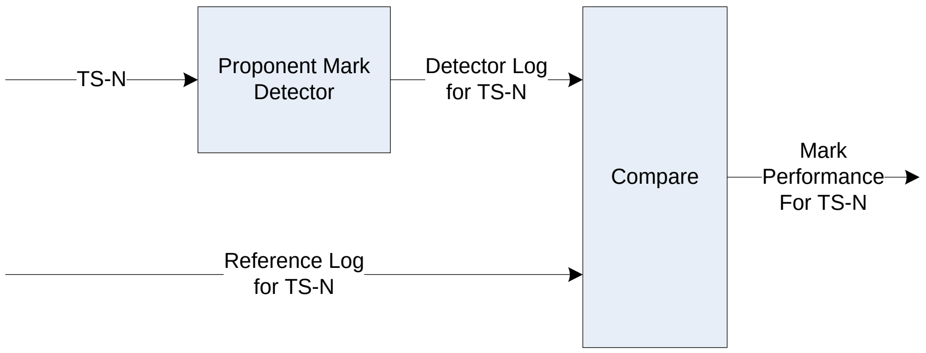


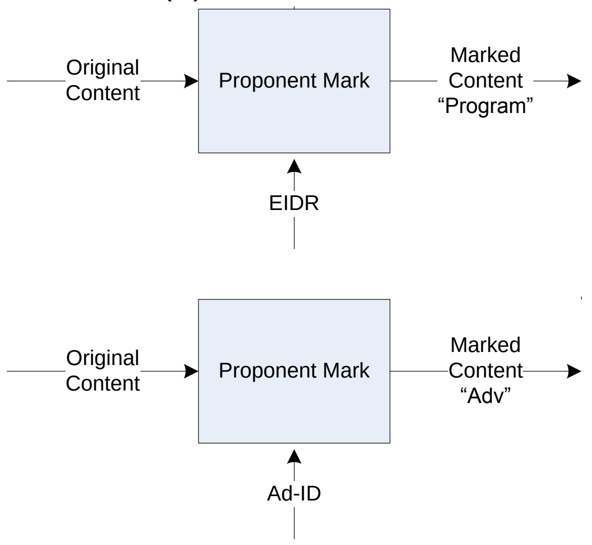
Figure 3 – Block Diagram of Detection and Performance Measurement Process

Two embedding scenarios are tested in the Robustness Test, which are described here and illustrated in Figure 4, below:

A. Embed EIDR in Content to produce "Marked Content – Program" and Embed Ad-ID in Content to produce "Marked Content – Adv", as in sub-figure labeled (A). The separate markings for EIDR and Ad-ID in effect double the amount of Marked Content and the two sets of Marked Content are used in constructing the Test Set. In this case the detector would be expected to detect either Ad-ID or EIDR MediaID values.

B. Embed EIDR in Content and then replace with new EIDR to produce Marked Content – "Program" and embed Ad-ID in Content and then replace with new Ad-ID to produce Marked Content – "Adv", as in sub-figure labeled (B). The separate markings for EIDR and Ad-ID in effect double the amount of Marked Content and the two sets of Marked Content are used in constructing the Test Set. In this case the detector would be expected to detect only the replacement EIDR MediaID value or the replacement Ad-ID MediaID value.

(A) – Mark Ad-ID for Adv; Mark EIDR for Programs



(B) – Replacement

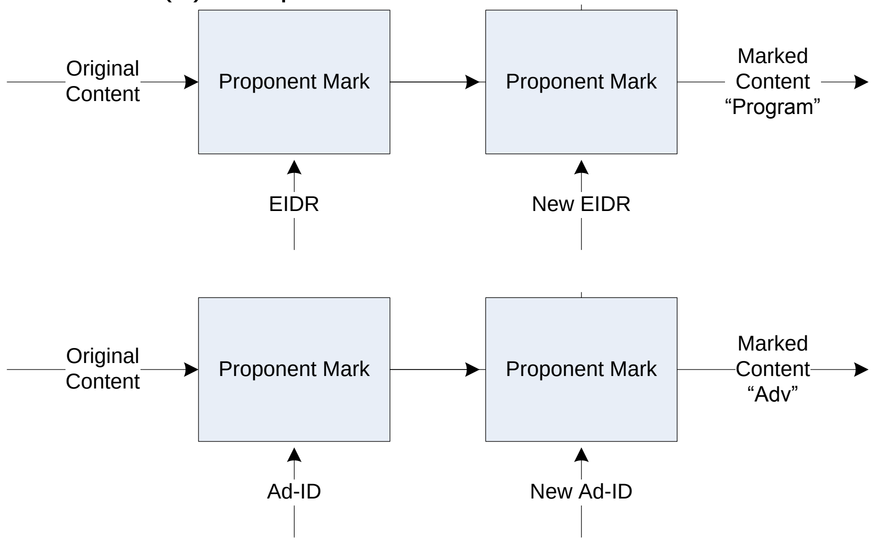


Figure 4 – Two modes for marking Content

Given that there are two marking scenarios and a number of audio processing scenarios, Table 1, below, illustrates how the resulting Test Sets are assigned unique names. Each marking scenario is described using three letters:

- Mrk – Marked with Ad-ID or EIDR
- Rpl – Ad-ID replaced by new Ad-ID and EIDR replaced by new EIDR

Processing chain is indicated by a single numeral, indicated by "N" in Table 1, below.

Table 1 – Example of Mark, Processing Chain and Test Set Nomenclature.

Mark	Processing Chain	Test Set Name
Mrk	PC-N	TS-Mrk-N
Rpl	PC-N	TS-Rpl-N

2.1 Terms

The following terms are defined:

General

- **Service -** A collection of media components presented to the user in aggregate such as a sequence of TV Programs in a continuous linear fashion. The term "channel" is used in this document as a synonym for "service."
- **Service time -** a media timeline that does not reset with each content change within a continuous Service.

Audio

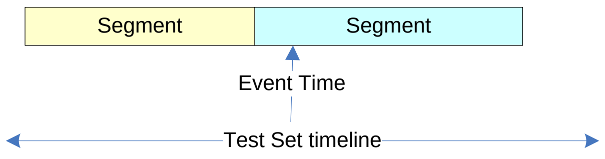
- **Audio Sampling Rate** – the number of samples per second, where samples can be individual samples in the case of a 1-channel audio signal or audio Sample Frames in the case of a multichannel audio signal.
- **Audio Sample Frame** – the samples at a single sampling instant from each of the channels in a multichannel audio signal. The samples in a Sample Frame have a specific order (e.g. see Annex A). An interleaved WAV file consists of a sequence of audio Sample Frames.

Information conveyed in Watermark

- **MediaID** – information identifying either an Ad-ID or EIDR or both, where the MediaID is associated with a specific portion of the Service, or Segment in terms of this test process.

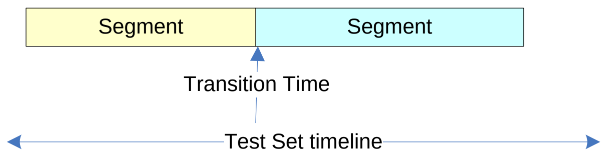
Test Set

- **Content** – Generic reference to any of Original Content, Marked Content, Processed Original Content or Processed Marked Content.
- **Original Content** – Content defined in Annex A.
- **Marked Content** – The Original Content marked by the proponent's embedder.
- **Media** – alternate name for Content.
- **Processing Step** – A processing step that would be found in the broadcast delivery chain, as defined in Annex B Table 5.
- **Processed Original Content** – Original Content that has been processed by a given Processing Chain, as defined in Annex B Table 5.
- **Processed Marked Content**– Marked Content that has been processed by a given Processing Chain, as specified in Annex B Table 7.
- **Segment** – An excerpt from the Processed Original Content or Processed Marked Content. Segments are marked with the same MediaID.
- **Test Set** – A sequence of Segments randomly selected from Processed Original Content or Processed Marked Content. A specific Test Set is identified by the ID of its associated Processing Chain, e.g. TS-1.
- **Splice** – The point at which two segments are joined, as in forming part of the Test Set. Also referred to as **Boundary** or **Transition**. A splice may include a Gap.
- **Gap** – a region of "zero" audio signal.
- **TimeStamp** – a time instant, formatted as SSSS.SS (in seconds, to the nearest hundredth), from the Test Set time line.
- **Event Time** – the time (as a TimeStamp on the Test Set timeline) at which the detector reports a significant event, e.g. the reporting of a BoundaryTime estimate and a MediaID value. This is illustrated in the following figure, where the event is reporting the detection of a transition from one Segment to another (each Segment shown in different color in the figure below). The Event Time may typically have some latency with respect to the Transition Time.

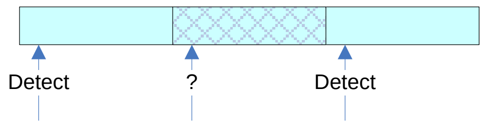


Performance

- **BoundaryTime** – The estimate of the time (as a TimeStamp on the Test Set timeline) that a transition from one Segment to another occurs (each Segment shown in different color in the figure below).



- **Identification Error** – a case in which the watermark detector incorrectly recovers the MediaID associated with a Segment.
- **Identification Discontinuity** – a case in which one Segment is reported as multiple separate Segments due to the false identification of a portion of the marked Segment as unmarked d (i.e. false negative) or differently marked. This is shown as the cross-hatched portion in the following figure:



2.2 Robustness Test Content

Audio Content shall be either 2.0 channels (stereo) or 5.1 channel signals, where the ".1" indicates LFE channel. All audio Content shall be interleaved WAV files sampled at 48 kHz with 24-bit word length. If Content is stereo, the stereo Content is marked and then placed into a 5.1 channel format where Left and Right are the stereo content and Center, Left Surround and Right Surround are digital zero. This is illustrated in the following figure.

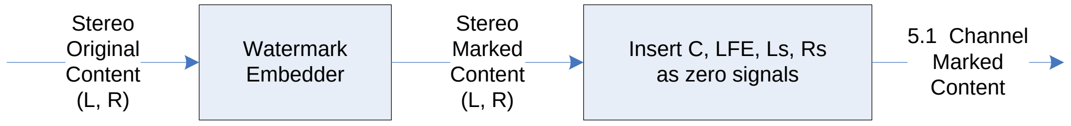

Stereo Original Content (L, R)
Watermark Embedder
Stereo Marked Content (L, R)
Insert C, LFE, Ls, Rs as zero signals
5.1 Channel Marked Content

Figure 5 –Processing for Stereo Content

2.3 Test Set Parameters

The following is a summary of parameter values used to construct the test set

Table 2 – Test Set Parameter Values.

Parameter	**Value**	**Description**
Num_Content_Files		Number of Content files, equal to number of rows in Table 3.
Test_Set_Len		Duration of Test Set, in hours. This depends on number and length of Content and Test Set parameters.
Segment_Len	3 to 15 seconds	Segment length for both Original and Marked Content, uniformly distributed between limits shown.
P_Gap	20%	Probability that a Segment will be followed with a Gap

Gap_Len	0.25-0.5 seconds	Gap Length, uniformly distributed between limits shown.
Num_Segments		This value can be derived from values above, and should be at least 1000.
P_Marked_Seg	2/3 (66.67%)	Probability that a Test Set Segment will be a marked Segment (as opposed to an unmarked Segment).
P_Marked_EIDR	50%	Probability that a Marked Test Set Segment will be a marked with EIDR (as opposed to Ad-ID).
P_Cue	15%	Probability that the next Segment will be a Cue (i.e. "skip ahead" or "skip back") within the current Segment, i.e. retaining the current MediaID.
MediaID_Header		8 bits of random data that is meant to indicate if MediaID is EIDR or Ad-ID.
MediaID – EIDR		A valid EIDR shall be constructed replacing "X" with random Hexadecimal digits in the following template: 10.5240/XXXX-XXXX-XXXX-XXXX-XXXX-C where the trailing "C" is a parity check character. The value of C used in the Robustness Test shall be set to a random character in the range [0-9 A-Z].
MediaID – Ad-ID		A valid Ad-ID shall be constructed according to http://www.ad-id.org/how-it-works/ad-id-structure

Random values in the table above shall actually be pseudo-random (and hence deterministic) as specified here:

- The EIDR values shall be constructed using a pseudo-random sequence generator such that the EIDRs embedded in the Content are deterministic.
- The Ad-IDs shall be constructed such that the set of possible systematic values are selected using a pseudo-random sequence generator such that the Ad-IDs embedded in the Content are deterministic.
- The MediaID_Header value shall be set using pseudo-random sequence generator such that the values are deterministic.

2.4 Overview of Test Set Construction

The Test Set is assembled in such a way as to approximate broadcast programs and viewer behavior, including channel changes, program changes within the same channel (e.g. break to advertisement) and media timeline changes caused by time-shifting (i.e., skip forward/back). In the case of channel changes there may be a small interval of no signal (or gap) that simulates the time required for a set top box to acquire the new channel signal. In the case of program changes within the same channel, such as the break to an advertisement, or PVR "cue" actions (e.g. skip forward or skip back), there may also be a gap that reflects splicing the content at e.g. the cable head end. Additionally, it is possible that not every channel or every content segment would be marked, so Processed Original Content and Processed Marked Content are both included in the Test Set.

The Test Set is constructed as follows:

- A number of files of Original Content are obtained. The exact selections and the duration of each are identified in Annex A.
- The proponent's watermark embedder marks the Original Content as described in Section 2 and Figure 4 to create the Marked Content according to the the marking scenarios. The output file must be the same size, (i.e. number of audio Sample Frames), same format (i.e. sampling rate and word size) as the input file and be time-aligned with the input file.
- The Original Content and the Marked Content are each subject to the processing steps in a specific Processing Chain to create Processed Original Content and Processed Marked Content. The processing steps that form a Processing Chain are shown in Table 7. The output file must be the same size, (i.e. number of audio Sample Frames), same format (i.e. sampling rate and word size) as the input file and be time-aligned with the input file.
- A Test Set is assembled from the Processed Original Content and Processed Marked Content. All proponents use the same parameters to assemble the Test Set (i.e. all proponents use the identical sequence of Segments, except for differences due to individual proponent watermarks). See Annex D for a description of Test Set creation scripts.
- Random segments from the Processed Original Content and Processed Marked Content are selected. These segments are Segment_Len in length for both Original and Marked Content. The Test Set assembly process should result in a Test Set with approximately Num_Segments transition points in the Test Set.
- Segments are concatenated to form the Test Set, in which
    - With probability P_Gap Segments are followed by a Gap of "zero audio" with a random duration of Gap_Len. This is meant to simulate a channel-change and the acquisition time for the new channel.
    - Some transitions shall simulate time-shifting on the STB, i.e., the Content and MediaID is the same across the transition, but the media timeline has a discontinuity (i.e. cue to skip ahead or skip back).
    - A Segment from any Content file and from either Processed Original Content or Processed Marked Content may follow any other Segment.
    - An audio Splice point is a Sample Frame.
    - The start of each Segment has a "fade-in" of the audio signal and the end of each Segment has a gradual "fade-out."
- A Reference Log will be generated by the Test Set assembly script. This Reference Log will capture the "ideal output" of a watermark system. See Section Annex C for details of Reference Log format.

3 Robustness Test Procedure

3.1 Detector Output

The purpose of the watermark is to identify the extent of broadcast content marked with a MediaID. In the Robustness Test this is corresponds to identifying the beginning and ending boundaries (as TimeStamps in the Test Set timeline) of Marked Segments that are marked with the same MediaID.

The proponent's watermark detector is run on each Test Set (e.g. TS-Mrk-1, TS-Mrk-2, etc.). For each Test Set, the proponent watermark detector shall output a Detector Log in the format shown in Annex C. These Detector Log files will be compared with the corresponding Reference Log file to determine the performance of the watermark system for a given Test Set. The Reference Log and Detector Logs for each Test Set may be included in a proponent's test report, and shall be retained by proponents in the event that SMPTE requires them in the future.

3.2 Performance Metrics

For each Test Set, proponents shall report the following metrics:

- Mean value and 10th and 90th percentile of the error in Boundary start time, as TimeStamp.
- Mean value and 10th and 90th percentile of the error in Boundary end time, as TimeStamp.
- Number and percentage of Marked Segments with miss-identified MediaIDs
- Number and percentage of Marked Segments with discontinuity. (This is a form of Marked Segment miss-identified MediaID.)
- Average Coverage for Marked Segments, in percent
- Average "Overage" for Marked Segments, in percent and in seconds (with 2 decimal digits)

Error in Boundary time is the difference between the Boundary start or end time reported in proponent detector log and that reported in the Reference Log, for each Boundary in the Test Set. Coverage and "Overage" metrics are defined with the aid of **Figure 6**, below. While Coverage is the extent that the detector correctly identifies the entirety of the marked Segment, Overage is the extent that the detector identifies as Segment boundaries times outside (either before or after) the true Segment boundaries. Referring to the figure, T_E is the duration of a Segment marked with EIDR, which begins at time T_1 and ends at time T_2.

If the detector identifies that the exact Segment start and Segment end boundaries (i.e. $T_A = T_B = T_C = T_D = 0$, then:

Coverage = 100%

Overage = 0 percent and 0.00 seconds

If the detector identifies that the Segment start boundary as $T_1 + T_B$, then the portion T_B is a shortfall in coverage. Similarly, if the Segment end boundary is identified as $T_2 – T_C$, then the portion T_C is a shortfall in coverage.

If the detector identifies that the Segment start boundary is $T_1 – T_A$, then the portion T_A is an Overage. Similarly, if the Segment end boundary is identified as $T_2 + T_D$, then the portion T_D is an Overage.

T_B and T_C can decrease to 0 (100% Coverage), but they can never go negative, as that is considered as Overage. Similarly, T_A and T_D can decrease to 0 (0% Overage), but they can never go negative, as that is considered as Coverage.

Hence, in general, Coverage and Overage can be computed as:

Coverage = $100*(T_E – T_B – T_C)/T_E$ percent.

Overage = $100*(T_A + T_D)/T_E$ percent and $T_A + T_D$ seconds.

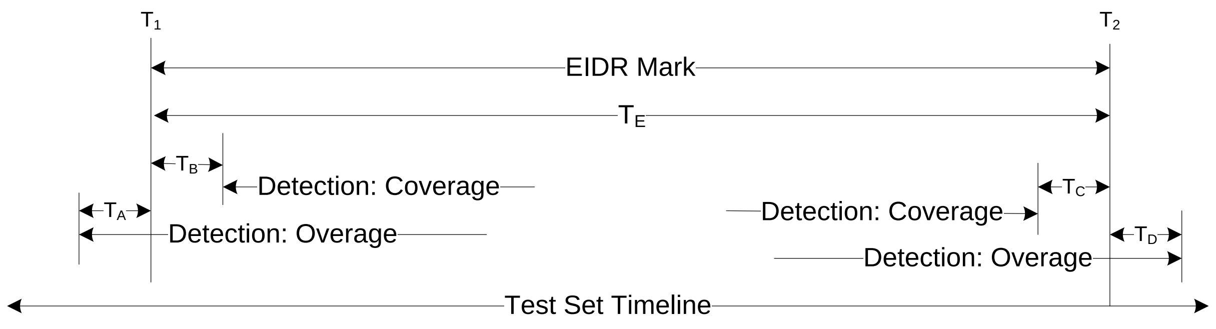


Figure 6 –Metrics based on Accuracy of Detection

4 Complexity Test Procedure

The Complexity Test measures the average CPU clock rate necessary to process specific Test Sets by a proponent’s watermark detector. The procedure minimizes the effect of operating system and file I/O overhead on the complexity measurement.

A common Reference Platform matching the specifications and settings stated below shall be used to ensure equal comparisons between proponents.[1]

Reference Platform Hardware (see also footnote 1 below)

- CPU: Intel i7- 4770, 4771 or 4790 running at 3400 Mhz
- RAM: 8 GB

[1] A proponent may opt to not use the Reference Platform and instead use a Substitute Platform that contains a different Intel CPU than the Reference Platform. Proponents using a Substitute Platform must perform all steps of the Complexity Test Procedure. If the Substitute Platform has a different CPU Clock Rate than 3400 MHz, the proponent shall use the alternative clock rate in step 6.

- OS: Windows 7, 64 bit
- The use of a GPU and other algorithmic processing hardware external to the Intel CPU core is prohibited

Reference Platform Settings [2][3]

- Turbo Mode disabled in BIOS
- Only one core enabled in BIOS
- Hyperthreading disabled in BIOS
- Other platform settings must ensure Turbo Mode is disabled, only one core is enabled, and Hyperthreading is disabled.

The Complexity Test Procedure requires proponents to measure the elapsed time to process

- Test Set TS-Mrk-2 (i.e. test set in which EIDR and Ad-ID are present and then processed by APC-2).
- TBD Original Content

with and then without the Watermark Detector Processing enabled. “Watermark Detector Processing” includes all processing after the Test Set has been read from file and unpacked into memory as if the data had been received from an HDMI stream in an ATSC 3.0 watermark detection device. Any filtering or mixing of the data after reading from the Test Set file shall be considered part of the watermark detector processing.

Any of the following three “Timing Methods” can be used to measure elapsed time:

- Windows command window execution using MSDOS “time” command:

[2] Proponents may be able to use the following alternative method to disable Hyperthreading and enable only one core:

- Enter `bcdedit /set onecpu on` in a command prompt window that has been opened using “Run As Administrator” and reboot the platform.
- Verify using the CPU-Z utility.
- To return the platform to its defaults, Enter `bcdedit /set onecpu off` in a command prompt window that has been opened using “Run As Administrator” and reboot the platform.

Verify using the CPU-Z utility.

[3] Proponents may be able to set their platform clock rate to a specific value using the following procedure:

- Go to **Power Options** and click **Change plan settings**
- Select **Change advanced power settings**.
- Scroll down to **Processor power management** and expand it.
- Expand **Minimum processor state,** change **On battery** and **Plugged in** to P%.
- Expand **Maximum processor state,** change **On battery** and **Plugged in** to P%.
- Click Apply and OK.
- Verify using the CPU-Z utility.

P% is a percentage of the CPU’s rated clock rate with turbo mode disabled. When P% = 100%, the CPU may enter turbo mode if it otherwise not disabled.

Put the following sequence in a .BAT file. Execute the .BAT file. Subtract the start time from the end time and express the result in seconds to the nearest 100th of a second.

```
time <nul

execute watermark detector

time <nul
```

- Cygwin command window execution using Linux "time" command when executing watermark detector. Round up the "real" result reported by the command in seconds to the nearest 100th of a second.
- Internal Timing. Use ftime(), time(), QueryPerformanceCounter(), RDTSC or similar timing function to measure the elapsed time from the beginning of the watermark detector program to its end. Express the result in seconds at least to the nearest 100th of a second.

Complexity Test Procedure:

1) Configure Reference Platform per required settings.
2) Use a Timing Method to measure the elapsed time to process the Test Set with Watermark Detector Processing disabled. Windows Process Priority for the detector may be set to High if desired. Repeat this step ten times and average the results to obtain a single elapsed time measurement.[4]
3) Use a Timing Method to measure the elapsed time to process the Test Set with Watermark Detector Processing enabled. Windows Process Priority for the detector may be set to High if desired. Repeat this step ten times and average the results to obtain a single elapsed time measurement. (See also footnote 4 below.)
4) Subtract the result from step 2 from the result of step 3 to obtain Elapsed Time.
5) Determine the duration of the Test Set in seconds.
6) Calculate CPU Clock Rate (MHz) * Elapsed Time (seconds) / Test Set duration (seconds.) This is the average speed the CPU clock needs to run to process the Test Set in real time. For the Reference Platform, CPU Clock Rate equals 3400 MHz.
7) Run the CPU-Z utility (available from www.cpuid.com) while processing the Test Set with Watermark Detector Processing enabled and obtain a screenshot of the "CPU" tab page. The page will demonstrate the platform settings. This should not be done in parallel with step 2 because it could distort the Elapsed Time measurement. Proponents may need to ensure that the Windows Process Priority for the detector is set to Normal.

[4] Proponents who need to increase the accuracy of their time measurements may process the Test Set multiple times within their detector program.

Proponents shall either include or exclude Robustness Test Detector Output Log generation for steps 2 and 3. Exclusion is preferable.

Proponent shall report
- The Platform used
- The Timing Method used
- The result of step 6 in MHz.
- An estimate of the Memory Resources required by the detector executable.
- The screenshot from step 7

5 Verification of Embedder and Detector

Proponents shall use the following means to document the embedder and detector configuration used in the Robustness test, which shall also be the same embedder and detector configuration used in a possible subjective quality test. Proponent shall supply as part of the Robustness report:

- Md5sum hash for embedder executable.
- Md5sum has for any embedder configuration file or command line and arguments. Alternatively, the actual configuration file or command line can be supplied.
- Md5sum hash for detector executable
- Md5sum hash for any detector configuration file or command line and arguments. Alternatively, the actual configuration file or command line can be supplied.
- Md5sum for each of the Marked Content files using marking scenario in either EIDR or Ad-ID are embedded in Content (i.e. scenario “Mrk”).

This information should permit a third party to obtain the proponent embedder and detector executable, verify that they are the correct binary files, run the embedder and verify the Marked Content files using marking scenario “Mrk.” A Test Set could be constructed from this and the proponent Robustness Test report results reproduced for that test set.

6 References

1. Request for Proposals – Open Binding of IDs to Media.

Annex A Test Content

All Robustness Test content shall be in the following format:

- Linear PCM RIFF WAV
- Sampling Rates: 48 kHz
- Bit Resolutions: 24 bits
- Channels: 2 or 6 (as 5.1) as interleaved samples
    - For stereo, interleave order is L, R
    - For 5.1, interleave order is L, R, C, LFE, Ls, Rs

Table 3, below, lists the content used in the Robustness Test. How to obtain the raw content and create the final content with the duration listed below is described elsewhere.

Table 3 - Audio Test Content

ContentID	Filename	Description	Duration H:MM:SS	Format	Channels
1	big_bucks_bunny.wav	Animation	0:10:00	AC-3, 448 kb/s, 48 kHz / 16 bits	5.1
2	elephants_dream.wav	Animation	0:10:00	AC-3, 448 kb/s, 48 kHz / 16 bits	5.1
3	tears_of_steel.wav	Animation	0:10:00	AC-3, 448 kb/s, 48 kHz / 16 bits	5.1
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					

ContentID	Clip Home Page	Download link
1	http://www.bigbuckbunny.org	http://mirror.bigbuckbunny.de/peach/bigbuckbunny_movies/big_buck_bunny_1080p_h264.mov
2	http://www.elephantsdream.org	http://video.blendertestbuilds.de/download.blender.org/ED/ED_HD.avi
3	http://tearsofsteel.org	http://download.blender.org/demo/movies/ToS/Surround-TOS_DVDSURROUND-Dolby%205.1.ac3

Audio is extracted from *.mov files as follows:

```
ffmpeg –I big_buck_bunny_1080p_h264.mov--ss 00:00:00--t 00:10:00--
acodec pcm_s16le--vn big_buck_bunny.wav
```

```
For big_buck_bunny.wav, append approximately 4 seconds of digital zero
to bring content up to 10.00 minutes in length.
```

Audio is extracted from *.avi files as follows:

```
ffmpeg  -i ED_HD.avi--ss 00:00:00--t 00:10:00--acodec pcm_s16le--vn
elephants_dream.wav
```

AC-3 is decoded to PCM as follows:

```
ffmpeg--i Surround-TOS_DVDSURROUND-Dolby-5.1.ac3--ss 00:00:00--t
00:10:00--acodec pcm_s16le tears_of_steel.wav
```

Annex B Processing Chains

B.1 INDIVIDUAL PROCESSING MODULES

Audio Codecs

Proponents should obtain the encoders and decoders for the codecs listed in Table 4.

Table 4 - Audio Codecs

Codec	Encoder	Decoder
Dolby AC-3	Minnetonka SurCode for Dolby Digital 5.1 (See note 2, below)	fFmpeg (See note 3, below)
MPEG-4 AAC, HE-AAC and HE-AAC v2	dBpoweramp Music Converter with dBpoweramp Codec Central m4a (See note 4 and Annex E, below)	fFmpeg (See note 3 and Annex E, below)
Dolby E	Minnetonka SurCode for Dolby E Encoder (See note 1, below)	Minnetonka SurCode for Dolby E Decoder (See note 1, below)

Notes:

1. Available at: http://www.minnetonkaaudio.com/index.php?option=com_content&view=article&id=48&Itemid=75&lang=en&phpMyAdmin=2f8ebc5437f1c9t59fb7cb7r58337
2. Available at: http://www.minnetonkaaudio.com/index.php?option=com_content&view=article&id=51&Itemid=80 (Version 2.2.9 only, which must be requested from Minnetonka support, as this correctly supports command line execution)
3. Available at: http://ffmpeg.zeranoe.com/builds/win32/static/ffmpeg-20131024-git-2f31b73-win32-static.7z (Windows platforms).
4. Available at: http://www.dbpoweramp.com/purchase.htm and http://codecs.dbpoweramp.com/codecs-new/dBpoweramp-Codec-m4a%20FDK%20(AAC)%20Encoder.exe (version R15 only, Windows platforms).

The codecs listed in Table 4 should be run with parameters as listed in Table 5.

Table 5 - Audio Codec Processing as specified by run-time parameters

ID	Codec Processing
A1	5.1 channel Dolby AC-3 encoding/decoding at 384 kbps with content-specific Dynamic Range Control applied at encoder and decoder per Table 6

A2	5.1 channel HE AAC encoding/decoding at 128 kbps, Spectral Band Replication applied, Parametric Stereo not applied
A3	5.1 channel LoRo Downmix to stereo, as specified in Section B.3.3**Error! Reference source not found.**
A4	Stereo HE AAC v2 encoding/decoding at 32 kbps, Spectral Band Replication applied, Parametric Stereo applied
A5	5.1 channel Dolby E encoding/decoding at 1.5 Mb/s.

For each Content file, as indicated by ContentID in Table 6, below, the Dolby AC-3 encoder shall use the DRC Mode indicated in the table.

Table 6–- Dolby DRC Modes

ContentID	**DRC Mode**
TBD	speech
TBD	film light
1, 2	film standard
TBD	music standard

Note that any delay due to audio encode/decode processing must be removed so that time-alignment with the unprocessed original signal is preserved. This can be implemented using a Matlab script or as a Bash script using aFsp utilities:
(see http://www-mmsp.ece.mcgill.ca/Documents/Software/Packages/AFsp/AFsp.html and http://www-mmsp.ece.mcgill.ca/Documents/Downloads/AFsp/).

Downmix

The Audio Downmixer is a script or application that has as arguments: an input filename of 5.1 channel input content and an output filename of stereo content. Upon execution, the downmixer shall produce an output content file containing the input content downmixed to stereo LoRo (Left only/Right only) format as defined by:

Lo = L + -3dB*C + -6dB*Ls

Ro = R + -3dB*C + -6dB*Rs

This can be implemented as a Matlab script or as a Bash script using aFsp utilities (see aFsp reference URLs above).

B.2 AUDIO PROCESSING CHAINS

The audio processing discussed in Section B.1 are assembled into processing chains as shown in Table 7:

Table 7 – Audio Processing Chains

ID	Description of Processing Chain	Comment
APC-1	Perform A1	Simplest processing chain, based on ATSC 1.0 broadcast.
APC-2	Perform A1, then A2.	High compression link to 5.1 channel TVs.
APC-3	Perform A1, then A3, then A4	Delivery of stereo over 3GPP network
APC-4	Perform A5, then A1	Post-production processing followed by emission coding.

B.3 INVOKING AUDIO PROCESSING

B.3.1 AC-3 audio

For codec processing A1, encode by executing SurCode as:

```
surcodeac3.exe /o Output.ac3 /l "Input_L.wav" /r "Input_R.wav" /ls
"Input_Ls.wav" /rs "Input_Rs.wav" /c "Input_C.wav" /lfe
"Input_LFE.wav" /dr 384 /sr 48000 /dn 31 /dp DRC
```

where:

Input_L.wav, *Input_R.wav*, *Input_Ls.wav*, *Input_Rs.wav*, *Input_C.wav* and *Input_LFE.wav* are the left, right, left surround, right surround, center, and LFE channel input audio files.

DRC is set to indicate the applicable DRC type indicated in Table 5 with: *N* for None, *FS* for Film Standard, *FL* for Film Light, *MS* for Music Standard, *ML* for Music Light, or *S* for *Speech*

Output.ac3 is the output filename

For codec processing A1, decode by executing FFmpeg as:

```
FFmpeg.exe -i input.ac3 -acodec pcm_s24le -ar 48000 -ac 6
output.wav
```

where:

input.ac3 is the AC-3 input filename
output.wav is the multichannel PCM output filename

B.3.1 HE AAC audio

For codec processing A2, encode by executing dBpoweramp Codec Central m4a (note that the executable from the “m4a FDK (AAC)” directory must be used) as:

```
fdkaac.exe -b 128000 -p 5 --ignorelength -S -o output.m4a
input.wav
```

where:

input.wav is the multichannel PCM input filename
output.wav is the AAC output filename

For codec processing A2, decode by executing FFmpeg as:

```
FFmpeg.exe -i input.m4a -acodec pcm_s24le -ar 48000 -ac 6
output.wav
```

where:

input.m4a is the AAC input filename
output.wav is the multichannel PCM output filename

B.3.2 HE-AAC v2 audio

For codec processing A4, encode executing dBpoweramp Codec Central m4a (note that the executable from the "m4a FDK (AAC)" directory must be used) as:

```
fdkaac.exe -b 32000 -p 29 --ignorelength -S  -o output.m4a
input.wav
```

where:

input.wav is the stereo PCM input filename
output.wav is the AAC output filename

For codec processing A4, decode executing FFmpeg as:

```
FFmpeg.exe -i input.m4a -acodec pcm_s24le -ar 48000 -ac 2
output.wav
```

where:

input.m4a is the AAC input filename
output.wav is the stereo output filename

B.3.3 Mixdown to Stereo

For processing A2, execute FFmpeg as:

```
FFmpeg.exe -i input.wav -acodec pcm_s24le -ar 48000 -af
"pan=stereo:FL=FL+0.707*FC+0.5*BL:FR=FR+0.707*FC+0.5*BR"
output.wav
```

where:

input.m4a is the multichannel PCM input filename
output.wav is the stereo PCM output filename

B.3.4 Dolby E audio

For codec processing A4, encode executing TBD

For codec processing A1, decode by executing TBD

Annex C Detector Log File Format

For each detected watermark segment, the following information shall be reported in the Detector Log file, where the file is in tab-separated text format (i.e. fields are delimited by tab characters).

`EventTime` – The time at which the detector reported an estimate of BoundaryTime, where the time is relative to the start of the Test Set and reported as TimeStamp.

`EventType` – Has the value 0 if reporting the boundary time for the start of a Segment, or 1 if reporting the boundary time for the end of a Segment.

`BoundaryTime` – The estimated time of the transition from the previous Segment to this Segment, where the time is relative to the start of the Test Set and reported as TimeStamp.

`MediaID_Header` – This is the value of the additional 8-bit field that, as of now, identifies the MediaID as EIDR or Ad-ID, so this shall be reported as either the string "EIDR" or "Ad-ID."

`MediaID` –The Ad-ID or EIDER information detected in this Segment, reported as a 12-character Ad-ID string or a 24-character Hexadecimal EIDR.

An example Detector Log file is show below. The columns shows EventTime, EventType, BoundaryTime, MediaID_Header and MediaID, respectively. Note that in the log, each Marked Segment is associated with two lines: a first with EventType = 0 which is the report of the TimeStamp associated with boundary at the start of the marked Segment, and a second with EventType = 1 which is the report of the TimeStamp associated with boundary at the end of the marked Segment.

```
137.91      0      137.06   EIDR     10.5240/XXXX-XXXX-XXXX-XXXX-XXXX-C
152.39      1      151.82   EIDR     10.5240/XXXX-XXXX-XXXX-XXXX-XXXX-C
152.82      0      152.24   Ad-ID    0123456789AH
160.38      1      159.81   Ad-ID    0123456789AH
```

Annex D Description of Scripts for Robustness Test Set Creation

The Matlab scripts are provided by Verance for use by SMPTE members or guests. Scripts require Matlab R2014a.

The scripts should be used with the following directory structure under the top directory `RobustnessTest`:

`Orig`	Original media files
`Mark_<MS>`	Marked Media files
`ProcOrig`	Processed Original media files
`ProcMark_<MS>_<PC>`	Processed Marked media files
`Scripts`	Location of these scripts
`TestSets`	Test Sets files

In the directory names above, `<MS>` is replaced by a label indicating marking scenario, and `<PC>` is replaced by a label indicating processing chain. This results in replication of e.g. the `Mark` folder for each of the marking scenarios.

The script API, as script calling format, is described here:

Script: `mk_test_set_params.m`
Call: `mk_test_set_params(pfile, lfile)`

`pfile` (output) is the filename of the Test Set parameters; argument is without any file extension. Parameters are output as tab separated values (pfile.txt)

`lfile` (output) is Reference Log file; argument is without any file extension. This is the output of the "perfect" watermark detector. File will have *.txt extension.

This script creates the parameters that fully describe the Test Set segments, segment splice points and possible "channel-change" gap.

The `pfile` has one line per segment containing the following tab-separated fields:

`BoundaryStart` is TimeStamp in Test Set timeline as SSSS.SS seconds
`BoundaryEnd` is TimeStamp in Test Set timeline as SSSS.SS seconds
`GapLen` is duration as SSSS.SS seconds
`MarkFlag` is
- 0 if Segment is from Original Content
- 1 if Segment is Marked Content

`ContentID` is the ID of the Content, indicated in column 1 of Table 3

`MediaStart` is TimeStamp in Content timeline as SSSS.SS seconds indicating the time of the start of the Segment in the Original or Marked Content indicated by MarkFlag and ContentID.

`MediaIDType`

0 if MarkFlag is 0
otherwise
1 if EIDR
2 if Ad-ID

`MediaID`

Space if MarkFlag is 0
otherwise
if MediaType == 1 then EIDR string
if MediaType == 2 then Ad-ID string

Script: `mk_all_test_sets.m`
Call: `mk_all_test_sets`
Function: Invokes all scripts in the necessary sequence to create all Test Sets.

Script: `mk_test_set.m`
Call: `mk_test_set(pfile, po_list, pm1_list, pm2_list, test_set)`

`pfile` (input) is the Test Set parameters file, without any file extension. An extension of *.csv is assumed.

`po_list, pm1_list, pm2_list` (inputs) are files that contain a list of file pathnames of processed original media files and processed marked media files, respectively. For the case of marking scenario “Mrk” file pm1_list contains filenames marked with EIDR and file pm2_list contains filenames marked with Ad-ID. For the case of marking scenario “Bth” pm2_list is never accessed. The files are specified without file extension and an extension *.txt is assumed. The proponent would typically construct these files by hand.

`test_set` is Test Set filename, without any file extension. An extension of *.wav is assumed. The script creates the test set from the processed original and processed marked media files based on information in the parameter file.

Script: `mk_report.m`
Call: `mk_report(pfile, lfile)`

`pfile` is the Test Set parameters file, without any file extension. An extension of *.txt is assumed.

`lfile` is a proponent’s detector output log file, without any file extension. An extension of *.txt is assumed.

The script computes the information required for the proponent robustness test report. It supports missed segments and split segments, but may not support all possible detector outputs.