A customizable open-source framework for measuring and equalizing e2e delays in shared video watching

Mario Montagud, Fernando Boronat
Universitat Politècnica de Valencia (UPV)
C/ Paraninf 1, 46730, Grau de Gandia, SPAIN
{mamontor@posgrado, fboronat@dcom}.upv.es

Pablo Cesar
Centrum Wiskunde & Informatica (CWI)
Science Park 123, 1098 XG, Amsterdam
P.S.Cesar@cwi.nl

ABSTRACT

Low-latency and media sync are essential requirements to enable interactive multi-party services, such as Social TV. In this work, we present an open-source and customizable framework that allows measuring end-to-end (e2e) video delays and provides support for different types of media sync, including Inter-Destination Media Sync (IDMS). This framework can be used by researchers to investigate the suitability of different techniques for optimizing the system performance in terms of e2e delays and media sync.

Author Keywords
Delay; Media Sync; IDMS; Social TV; Clock Sync.

ACM Classification Keywords
C.2.4 [Communication Networks]: Distributed Systems

MOTIVATION

Social TV, in conjunction with other forms of shared media experiences, are gaining momentum [1]. To accommodate this transition towards *"networked togetherness"* around media content, low-latency and synchronized services must be provided. On one hand, end-to-end (e2e) delays and delay variability, both within individual streams and between different streams, can impair the interactivity with the media content. On the other hand, e2e delay differences between the involved users can spoil natural and coherent interactions between them. The process of compensating the e2e delay variability across separated devices is typically known as Inter-Destination Media Sync (IDMS).

Previous studies have revealed that the magnitudes of e2e delay differences in actual delivery systems ([2, 3]) are much larger than acceptable limits in typical IDMS use cases [1]. Accordingly, research must be targeted on devising underlying mechanisms to seamlessly palliate both the effects of e2e delay and delay variability. Both issues are tightly coupled, since being able to accurately measure and optimize e2e delays is a first step towards devising advanced IDMS solutions to equalize them.

TVX'14, June 25 – June 27, 2014, Newcastle, UK.
DOI.10.6084/m9.figshare.1032656

In this work, we present an open-source and easily extensible framework that allows automatically measuring capture-to-render (e2e) video delays in customizable media delivery scenarios. Unlike other proposed solutions (e.g., [3-5]), our measurement system does not require any users' involvement and it is fully integrated into the media framework, in which a full control on all involved components is available. Moreover, our framework provides support for enabling the different types of media sync (including IDMS) [6]. It can be used to help answering the following research questions (among others):

1. How (accurately) can e2e video delays be measured?
2. What is the impact of different sources of delay in the e2e video delivery chain? How can different components be chosen or tuned to optimize these e2e delays?
3. How much e2e delay (differences) is (are) noticeable or annoying in interactive (shared) video streaming scenarios?
4. How (much) can e2e delay differences be compensated? Is it feasible to deploy IDMS solutions in real scenarios?
5. Which strategies and techniques are best suited for improving the QoE when performing media sync (IDMS)?

PROTOTYPE

The prototype has been implemented using GStreamer. We have checked its standard-compliant behavior by analyzing the streams in Wireshark and by playing the media using other frameworks, such as VLC and MPlayer. The server and client/s architectures are shown in Figures 1 and 2.

Media Sync Support

Our prototype provides support for different types of media sync (i.e., intra-stream, inter-stream and inter-sender sync) by relying on the capabilities of RTP/RTCP (RFC 3550), as described in [6], as well as on proper buffering strategies. Most importantly, the above functionalities are extended with the following modules to also provide support for IDMS (see Fig. 3):

- RTSP Server (RFC 2326) with multicast support. It also allows for "sharing" (live) media between multiple clients, which can join the session at different instants.
- Clock Sync. It can be achieved by using 2 alternatives. The first one is using NTP (RFC 5905). The second one consists of using two useful GStreamer components: *i) NetTimeProvider*, which exposes a "master" wall-clock

on the network; *ii) NetClientClock*, which subscribes and gets enslaved to that "master" wall-clock.

- SDP Module (RFC 4566). Apart from the standard SDP capabilities, it allows providing the *wall-clock settings* and the target *"playout time"* for the initial RTP packet to each new client, thus enabling rapid sync.

Automatic e2e delay measurements

Our system is able to measure the capture-to-render delay for each video frame. For that purpose, at the server side, we use a GStreamer component, called *videomark*, which allows overlaying a barcode into each video frame. This barcode allows inserting a 64-bit integer value, which will include a NTP-based timestamp. This component is placed just after capturing/retrieving each frame (Fig. 1). At the client side, we use another GStreamer component, called *videodetect*, which is responsible of decoding the timestamp inserted into the barcode. The *videodetect* component is placed just before rendering each video frame (Fig. 2). This way, by comparing the capturing and rendering NTP-based timestamps, our prototype is able of automatically measuring (and logging) the e2e video delays.

Visually checking the e2e delay and IDMS performance

Our prototype additionally allows for visually checking the e2e delay and IDMS performance. This is achieved by overlaying numeric timestamps in each captured/rendered video frame and by launching snapshots, either in a user-transparent way when specific internal conditions are met or in a manual way by pressing a button (Fig. 4).

FUTURE WORK

We plan to keep improving this framework and use it in our research on several topics, especially on assessing the suitability of different strategies for media sync and their implications on the QoE. A complete documentation, examples, demo videos and the source code will be available at: https://sites.google.com/site/mamontor/.

ACKNOWLEDGMENTS

This work has been funded, partially, by UPV under its R&D Support Program in PAID-01-10 Project and by CWI under EU/FP7 REVERIE Project (ICT-2011-7-287723).

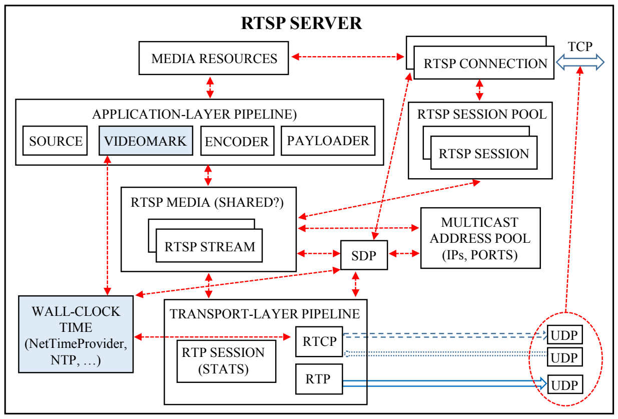


Figure 1. Server Architecture.

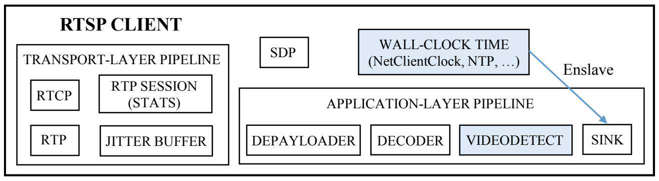


Figure 2. Client/s Architecture.

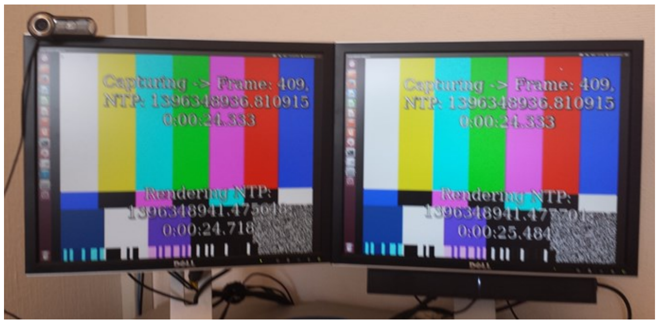
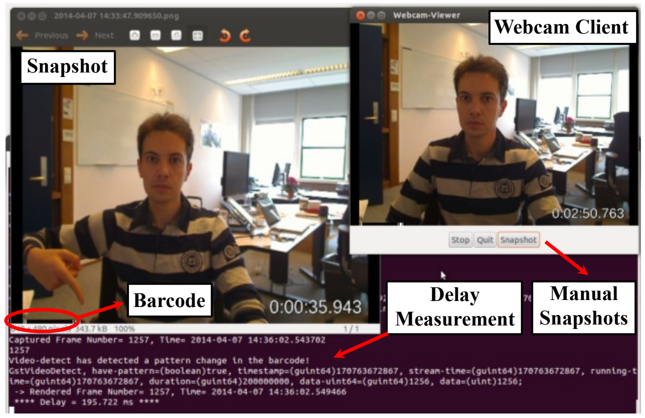


Figure 3. Synchronized Playback across Devices.



```
Captured Frame Number= 1257, Time= 2014-04-07 14:36:02.543702
1257
Video-detect has detected a pattern change in the barcode!
GstVideoDetect, have-pattern=(boolean)true, timestamp=(guint64)170763672867, stream-time=(guint64)170763672867, running-t
ime=(guint64)170763672867, duration=(guint64)200000000, data-uint64=(guint64)1256, data=(uint)1256;
 -> Rendered Frame Number= 1257, Time= 2014-04-07 14:36:02.549466
 **** Delay = 195.722 ms ****
```

Figure 4. Time Stamped Barcode, Snapshots Launching and e2e delay measurement.

REFERENCES

1. Montagud M., et al. Inter-Destination Multimedia Synchronization; Schemes, Use Cases and Standardization, *MMSJ*, 18(6), 459-482, Nov. 2012.
2. Boronat F., et al. Distributed media synchronization for shared video watching: Issues, challenges and examples, *Social Media Retrieval*, Computer Communications and Networks (Springer), pp. 393–431, 2013.
3. Woiter K., Playout Delay of TV Broadcasting, Master Thesis, University of Twente & TNO, 2014
4. Jansen J., et al. User-centric video delay measurements, *ACM NOSSDAV 2013*, Oslo (Norway), February 2013.
5. Kryczka A., et al. AvCloak: A Tool for Black Box Latency Measurements in Video Conferencing Applications, *IEEE ISM 2013*, California, Dec. 2013.
6. Montagud M., Boronat F. RTP/RTCP and Media Sync: A Review and Discussion of Future Work, *MediaSync Workshop 2013*, Nantes (France), October 2013.