https://doi.org/10.26552/com.C.2016.1.11-16

Oliwia Komperda - Hugh Melvin - Peter Pocta *

A BLACK BOX ANALYSIS OF WEBRTC MOUTH-TO-EAR DELAYS

Due to the recent shift toward cloud based computing, some of the world's leading standardization bodies have combined forces to provide guidelines and standards for native implementation of RealTime Communication (RTC) in the browsers. The World Wide Web Consortium (W3C) works on WebRTC APIs [1], while the Internet Engineering Task Force (IETF) is concerned with underlying standards [2]. Their efforts led to the development of WebRTC project by Google, Mozilla and Opera, which is an open framework that allows browsers to be RTC ready [3]. In this paper, we examine WebRTC from a Quality of Service (QoS) perspective, focusing on Mouth-to-Ear (M2E) delays under various application configurations.

Keywords: *VoIP, WebRTC, Skype, Mouth-to-ear M2E delay.*

1. Introduction

VoIP uses IP based networks (either public or private) in order to transmit digitized voice between two or more endpoints in real-time. The development of VoIP has eased the cost constraints of long-distance communications as well as completely revolutionized the way we think about real-time communications (RTC). Modern day RTC provide a rich variety of functions including, but not limited to multi-point voice calls, video conversations or data and screen sharing.

In May 2011 Google released a product called WebRTC under an open-source license. From that point, some of the world's leading standardization bodies have been combining forces to provide guidelines and standards for native VoIP support within browsers. The World Wide Web Consortium (W3C) works on WebRTC APIs, which define JavaScript APIs and HTML elements necessary to develop WebRTC communications. The Internet Engineering Task Force and their RTCweb (Real-time communications in WEB-browsers) team are concerned with underlying standards. The WebRTC project by Google, Mozilla and Opera is an open framework that allows browsers to be RTC ready. With only just a few lines of JavaScript code, developers can set-up a Video or Voice over IP (VoIP) sessions. The aim of this research is to quantify M2E delays introduced by WebRTC applications under varying application configuration settings. Firstly, baseline M2E delays are measured using standard WebRTC settings (i.e. Opus Codec with 20 ms packet size and 48000 sampling rate). Once the baseline delays are established, the impact of various Opus settings – namely packet size and encoding rate as well as other WebRTC provided codecs is examined.

The rest of the paper is organized as follows: In Section 2 the importance of mouth to ear delay (M2E) as a QoS metric is discussed. Section 3 briefly reviews WebRTC. Section 4 and 5 describe the experimental test-bed and experimental results. Finally, Section 6 concludes the paper and suggests some future work.

2. Importance of M2E delay as QoS metric

One of the most important aspects of Real-Time communications is Quality of Service (QoS). The internet was not designed with real-time capabilities in mind, therefore QoS support is required for some applications. This may be achieved in several ways including application and network configuration. Due to the time sensitivity of the voice packets, QoS is of exceptional importance when talking about VoIP applications, including WebRTC. Three main QoS indicators of the performance of VoIP applications are as follows: network jitter, packet loss rate and mouth-to-ear delay [4]. At the sender side, delays include encoding and packetization. These will depend on factors such as codec and frame size used. Other delays on sender side include operating system, sound card and NIC serialization delay. While being transmitted over the network, packets are subject to congestion delays at all intermediate routers as well as baseline propagation and serialization delay. At the receiving point packets experience similar delays to those at the sending

* [1]Oliwia Komperda, [1]Hugh Melvin, [2]Peter Pocta
[1]Discipline of Information Technology, College of Engineering & Informatics, National University of Ireland, Galway, Ireland
[2]Department of Telecommunications and Multimedia, Faculty of Electrical Engineering, University of Zilina, Slovakia
E-mail: o.komperda1@nuigalway.ie

point, as well as extra delays caused by jitter buffer [5, 6 and 7]. The ITU Telecommunication Standardization Sector (ITU-T) provides some guidelines in the area of telecommunications. More specifically, ITU-T Recommendation G.114 [8] for mouth-to-ear delay outlines that delays of 0-150 ms are acceptable to most users, delays of 150-400 ms are acceptable, but will impact on the quality of call, while delays of over 400 ms are generally unacceptable. Those values are only applicable if relevant echo management techniques are employed.

Delays can have a drastic impact on the interactive nature and thus the QoS for VoIP. More generally, there is a growing awareness of the importance of deterministic timing for many diverse application areas, including real-time communication (RTC). To achieve such temporal determinism, research is necessary in many areas in order to create so-called Time-Aware Applications, Computers and Communication Systems (TAACCS). TAACCS is a recently established interest group that have identified areas requiring research as follows: clock and oscillator designs, time and frequency transfer methods, the use of timing in networking and communications systems, hardware and software architecture, design environments, and the design of applications. Current systems use timing mostly as performance metric. In order to provide higher QoS in the area of telecommunications, better time and timing awareness between hardware, software and the network is thus necessary. Further details on the TAACCS project can be found at [9].

3. WebRTC

WebRTC is an innovative approach to real-time communications. Its target is to implement real-time communication functionalities into all browsers making them accessible to developers through HTML and JavaScript. Some of the major international standardization communities are currently working on standards and guidelines for implementation of WebRTC into browsers [10]. Those standards are already being implemented by some of the browser vendors. WebRTC introduces a concept of peer-to-peer streaming into web browsers. In this new model two browsers are able to communicate directly between each other once the Session Description Protocol (SDP) offer has been negotiated. The Signaling Server is used to provide a signaling channel between the ends of the peer-to-peer connection. Communication is done using on-the-wire standard protocols, which normally use User Datagram Protocol (UDP) for transport. Real-Time Transport Protocol (RTP) and Real-Time Transport Control Protocol (RTCP) are used to provide more reliability when transporting time sensitive data over UDP [11].

4. Test Design

M2E delay is one of the crucial factors of QoS. Identifying the extent of delays introduced by WebRTC application could help pinpoint flaws in the current WebRTC implementation and identify areas of the project that require improvements. The questions addressed by this paper are as follows:

- How do various codecs impact on WebRTC M2E delay?
- For Opus, how does packetization size impact on WebRTC M2E delay?
- For Opus, how does encoding rate impact on WebRTC M2E delay?

A local server was used to enable a peer-to-peer WebRTC connection. Ping was used to estimate network RTT delay both in advance and during tests as our primary interest was in application performance. The SDP was used to modify the various application settings. An oscilloscope attached to both input and output acoustic interfaces ensured the most accurate results possible. The test bed is depicted in Fig. 1.

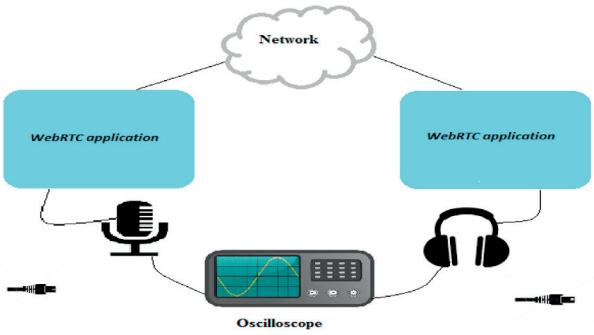


Fig. 1 Test bed

For each media/test-bed configuration, 10 VoIP calls were executed. Each call lasted between three and three and a half minutes. In each call duration, ten identifiable signals were sent between the peers, each approximately 20 seconds apart. Before and throughout each call, a ping signal was sent from one peer to the other. For each test configuration, one hundred measurements (10 x 10) and hundred ping signals (10 x 10 – 10 seconds interval) were taken for each media configuration. Two identical desktop computers were used, connected via a well provisioned (100 Mbps) wired LAN. Their specifications are as follows: Processor: AMD Athlon™ 64 X2 Dual Core Processor 4400+ 2.30 GHz. RAM: 4.00 GB, OS: Windows 7 Enterprise (64-bit). For browser, Google Canary Chrome (Version 39.0.2171.95 m) has been used.

5. Results

A. Baseline

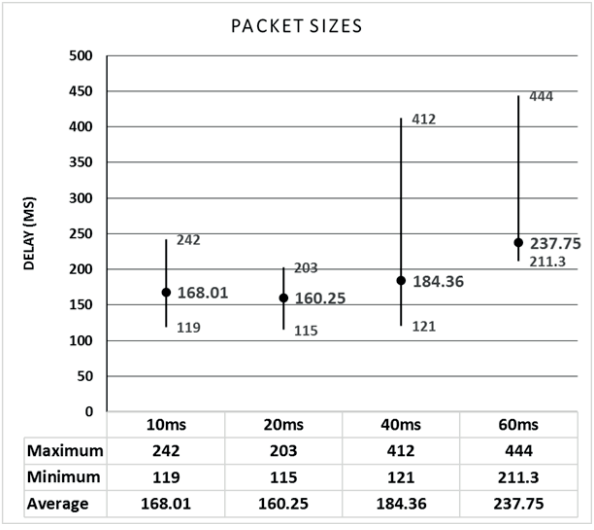
Firstly, M2E baseline delays for connections involving the default Opus codec were captured. It should be mentioned that ping results for this configuration were negligible with less than 2 ms round trip delays. Table 1 summarizes delays, outlining average and standard deviation. The experiment was performed using Voice only and then with Voice/Video call configuration. The results illustrate that in absence of significant network delays, the WebRTC default application produced M2E delays that approach the G.114 limit levels.

Summary of baseline delays Table 1

Application	Average m2e delay (MS)	Standard Deviation
WEBRTC VOICE + VIDEO	155.43	22.4235
webrtc Voice only	144.46	8.557093

B. Packetization

The influence of packetization size on M2E delay for Opus codec was then examined. Figure 2 shows average delays for 10, 20, 40 and 60 ms packets respectively, as well as the full range for each packet size. For each call, ten delay samples were taken. The Ping facility, as above, was used to measure network related delays. For all of the packet size tests, ping was negligible with values never exceeding 1ms.



	10ms	20ms	40ms	60ms
Maximum	242	203	412	444
Minimum	119	115	121	211.3
Average	168.01	160.25	184.36	237.75

Fig. 2 Packet size delays

Interestingly, a non-linear increase in M2E delays was observed as packet sizes increase, as shown in Fig. 2. Increasing packet size from 20 ms to 40 ms results in average M2E delay increase of 24 ms. This seems acceptable as an additional 20 ms of delay are added by increasing length of voice sample from 20 to 40 ms, the remaining 4 ms are possibly added due to longer processing requirements caused by encoding and transmitting of bigger packet. However – a further increase of packet size to 60 ms results in an additional 53 ms of delay - from 184 to 237 ms. Interestingly, reducing packet size from 20 ms to 10 ms did not result in M2E delay decrease. This could be caused by protocol headers overhead and the way in which WebRTC library handles 10 ms packet sizes for Opus codec. As evident from Fig. 2, a greater variation between single M2E delays for 10ms packets has been observed in comparison to that achieved for 20 ms packets.

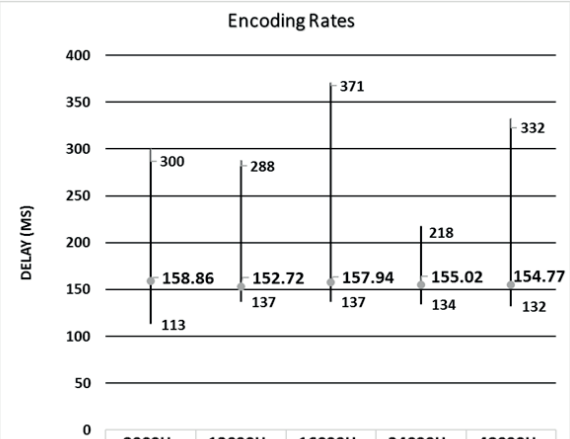
Again, in the context of G.114, it is interesting to note the non-linear impact and the extent to which application settings can greatly impact on M2E delay. In the absence of forensic under-the-hood analysis of the WebRTC codebase, the precise reasons for this non-linearity are not known. In section E below, we identify the jitter buffer behaviour as being the main contributor to this increase but also rule out sender side jitter in sending interval as a primary cause. Whilst such analysis is ongoing, we can speculate that issues such as drivers can be a contributory factor. Previous research by one of the authors has shown very unusual M2E behaviour casued by a mismatch between driver and application settings [5]. As outlined above, this reinforces and highlights the more general concerns raised by the TAACCS interest group and the consequent need for better Time Awareness.

C. Sample Rate

Besides deploying different packet sizes, Opus can also operate with a wide range of sampling rates. The sampling rate can be defined as a number of samples taken for each second of the signal. It is important to remember that Opus is a variable bit-rate codec. Bit-rate is a number of bits sent over the network in each second of the connection. Generally as sampling rate increases, greater bitrate is required to transmit data. That means that depending on network conditions and current performance, Opus can adjust encoding rate in order to maximize a quality of call [12]. The encoding rate will never exceed the maximum value as passed in SDP offer, however it may be set up to any value below it during the duration of call. Tests were performed to measure M2E delays under five encoding rates: 8000 Hz, 12000 Hz, 16000 Hz, 24000 Hz and 48000 Hz. The ping values again indicated negligible network delay with values between 0 and 3 ms.

As shown in Fig. 3, encoding rates did not have a noticeable impact on M2E delays. As expected, the bitrates present during the calls increase slightly as encoding rates rise. Opus supports bitrates of up to 510 kbps, however encoding bitrates of up to 48

kbps are mostly used for voice. During our tests, values of up to 45 kbps were observed. It is important to note that the quality of voice speech encoded at bitrates of up to 48 kbps is still very high [13].



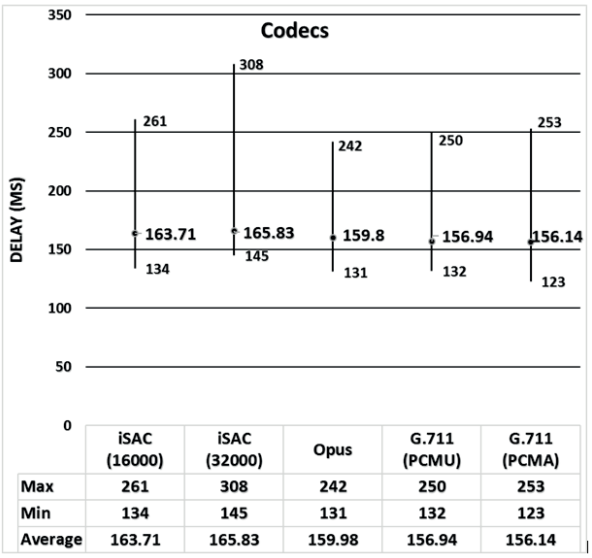
	8000Hz	12000Hz	16000Hz	24000Hz	48000Hz
Maximum	300	288	371	218	332
Minimum	113	137	137	134	132
Average	158.86	152.72	157.94	155.02	154.77

Fig. 3 Encoding Rate Delays

D. Codec

Besides Opus, WebRTC supports a number of voice codecs. Both Opus and G.711 are mandatory to implement for any WebRTC solution. This is to make sure that all the endpoints of the connection support at least one common codec. The browser generates an SDP offer contains a list of all the codecs supported on a given machine. In this way, the end points negotiate at the beginning of the communication, by comparing their SDP offers, and agreeing on a codec for voice communication. Tests were performed between two endpoints to measure M2E delays for various codecs supported by WebRTC library. Ten calls were performed for each codec and ten samples were taken during each call. Before each sample a Ping signal was sent between two endpoints to measure network delay. As before, Ping values for those tests never exceed 1 ms. In this test, the following codecs were tested: iSAC operating at 16 kHz, iSAC operating at 32 kHz, Opus operating at 48 kHz, G. 711 A-law and G.711 *μ-law operating at 8 kHz. The packet size for each codec is following: 30 ms (iSAC (16000)), 30 ms (iSAC (32000)), 20 ms (Opus), 20 ms (G.711 (PCMU)) and 20 ms (G.711 (PCMA)) respectively.*

As shown above in Fig. 4, the delays imposed by various codecs are relatively very similar, with iSAC codec shows slightly higher delays than Opus and G.711 for both sampling rates. That could be explained by the fact that iSAC operates on speech frames involving 30 ms of speech samples as oppose to speech frames of length of 20 ms used for Opus and G.711. Although the average



	iSAC (16000)	iSAC (32000)	Opus	G.711 (PCMU)	G.711 (PCMA)
Max	261	308	242	250	253
Min	134	145	131	132	123
Average	163.71	165.83	159.98	156.94	156.14

Fig. 4 Codec delays

values of M2E delay are quite similar for both Opus and iSAC, the delay range for iSAC is also greater. Opus codec achieved slightly higher average values to those reported for G. 711, however minimum and maximum values of delays for both codecs suggest that delays imposed by both of them are of the same range. Moreover it is important to note that Opus as a wideband codec is much superior to G.711 (a narrowband codec) in terms of quality of experience. Moreover G.711 performs almost identical for both versions, namely A-law and *μ-law*.

E. Jitter Buffer

The non-linear increase in M2E delay for different packet sizes outlined above encouraged the authors to investigate the jitter buffer behaviour during those calls. It is worth noting here that the jitter buffer values can be observed using chrome://webrtc-internals/. Our observations showed that when packet size is changed from the default value of 20 ms, the jitter buffer starts introducing additional delays, see Figs. 5 - 8 for more detail.

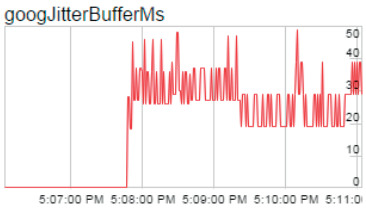


Fig. 5 Jitter buffer for 10 ms packet size

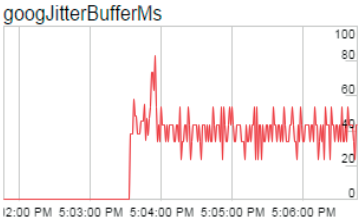


Fig. 6 Jitter buffer for 20 ms packet size

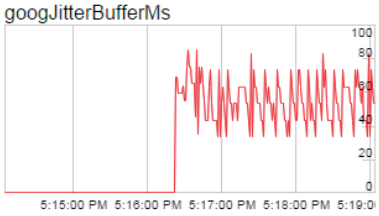


Fig. 7 Jitter buffer for 40 ms packet size

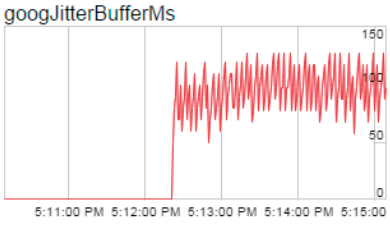


Fig. 8 Jitter Buffer for 60 ms packet size

We initially speculated that one possible reason for such jitter buffer behaviour could be irregular transmission of packets at the sending side caused by sender side non-determinism. The authors investigated the sender-side time intervals for each packet size using the Wireshark network analyzer. We concluded that with exception of a few out of place packets, packets are being transmitted relatively regularly from the sending side, regardless of packetization. There is some jitter in the data - however as the issue occurs for all the packet sizes, it is clear that sending time intervals are not the source, and thus the cause of jitter buffer behaviour lies elsewhere. We plan to investigate this issue as a further step in this research.

6. Conclusions and further work

Overall, some very interesting results in terms of M2E delays of WebRTC have been presented in this paper. The results firstly showed that baseline delays for WebRTC in presence of minimal network delay and jitter (treated as negligible) resulted in delays very close to or greater than those defined in ITU-T Rec. G.114. This is of significant concern for QoS/QoE as network delay can easily introduce 10-100 ms baseline increase of delay with jitter adding further due to jitter buffer behaviour. Regarding other tests, the sampling rate and in turn bitrate for Opus codec has no significant impact on M2E delays that occur during WebRTC calls. However, the experiments with packet sizes showed its significant impact on M2E delay. The packet size of 20 ms has been identified as the preferable choice, as it resulted in the lowest M2E delays. For a packet size increase to 60 ms, the corresponding M2E delay increase is of particular note. Moreover the operation of jitter buffer has been identified as a source of additional delay. While the jitter buffer for 20 ms packets tends to stay within 30-50 ms range, once the packet size increases to 60 ms jitter buffer values increase to between 60 and 130 ms. The source of such behaviour was not evident from an analysis of sender side packets, and further research is needed.

The study also showed that codec choice had little impact on M2E delay. Delays introduced by iSAC codec for both sampling rates were slightly higher than those achieved for Opus as well as G.711 though packetisation size is the obvious reason for this.

In conclusion, the tests emphasize the need for RTC application developers to ensure Time Awareness in the design and implementation of RTC applications. More broadly, these test results highlight the essential message of the TAACCS project whereby the full chain of hardware and software need to be better integrated and Time Aware so as to provide better guarantees and temporal determinism.

Acknowledgement

This work has been partially supported by the ICT COST Action IC1304 - Autonomous Control for a Reliable Internet of Services (ACROSS), November 14, 2013 - November 13, 2017, funded by European Union.

References

[1] W3C, World Wide Web Consortium, no date, accessed on 04/12/14 from http://www.w3.org/

[2] IETF, International Engineering Task Force, no date, accessed on 03/12/14 from https://www.ietf.org/

[3] WebRTC.org, WebRTC project, accessed on 09/10/14 from www.webrtc.org

[4] KARAPANTAZIS, S., PAVLIDOU, F-N.: VoIP: A Comprehensive Survey on Promising Technology, *Computer Networks*, 2009, vol. 53(12), pp.2050-2090, accessed on 18/10/14 from: http://www.sciencedirect.com.libgate.library.nuigalway.ie/science/article/pii/ S1389128 609001200

[5] MELVIN, H.: *The Use of Synchronized Time in Voice over Internet Protocol (VoIP) Applications,* Unpublished doctoral dissertation, University College Dublin, accessed on 16/11/15 from http://www.csi.ucd.ie/staff/jmurphy/publications/thesis-hugh.pdf, 2004

[6] MARKOPOULOU, A. P., TOBAGI, F. A., KARAM, M. J.: *Assessment of VoIP Quality over Internet Backbones,* Proc. 2002, vol. 1, pp. 150-159, accessed on 18/01/15 from http://ieeexplore.ieee.org.libgate.library.nuigalway.ie/xpl/articleDetails.jsp?arnumber=1019256

[7] CISCO.COM: Understanding Delay in Packet Voice Networks, February 2006, accessed on 24/11/2014 from http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/5125-delay-details.html

[8] ITU-T, Recommendation G.114: One way transmission time, 2003, Accessed on 07/11/14 from http://www.itu.int/rec/T-REC-G.114-200305-I/en

[9] WEISS, M., EIDSON, J., BARRY, CH., BROMAN, D., GOLDIN, L., BOB IANNUCCI, LEE, E. A., STANTON, K.: *Time-Aware Applications, Computers, and Communication Systems (TAACCS),* NIST Technical Note 1867. NIST, National Institute of Standards and Technology, http://dx.doi.org/10.6028/NIST.TN.1867, U.S. Department of Commerce, February 2015.

[10] ALVESTRAND, H.: *Overview: Real Time Protocols for Browser-based Applications Draft-ietf-rtcweb-ovieview-13*, November 28 2014, accessed on 12/02/15 from https://tools.ietf.org/html/draft-ietf-rtcweb-overview-13

[11] JOHNSTON, A. B., BURNETT, D. C: *WebRTC, APIs and RTCWEB. Protocols of the HTML5,* Real-Time Web, Digital Codex LLC, St. Louis, 2014

[12] VOS, K., VALIN, J.: RTP *Payload Format for Opus Speech and Audio Codec Draft-ietf-payload-rtp-opus-11*, April 2015, accessed on 20/05/15 from https://tools.ietf.org/ html/draft-ietf-payload-rtp-opus-11

[13] VALIN, J.-M., TERRIBERRY, T. B, MONTGOMERY, C., MAXWELL, G.: A High Quality Speech and Audio Codec with Less Than 10ms Delay, *IEEE Transaction on Audio, Speech, and Language Processing*, vol. 18, No. 1, 2010, accessed on 01/07/15 from http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4926218.