Albert Abelló Lozano

Performance analysis of WebRTC

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 20.3.2012

Thesis supervisor:

Prof. Jörg Ott

Thesis advisor:

M.Sc. (Tech.) Varun Singh

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Albert Abelló Lozano

Title: Performance analysis of WebRTC

Date: 20.3.2012 Language: English Number of pages:4+6

Department of Communication and Networking

Professorship: Networking Technology Code: S-55

Supervisor: Prof. Jörg Ott

Advisor: M.Sc. (Tech.) Varun Singh

Your abstract in English. Try to keep the abstract short, approximately 100 words should be enough. Abstract explains your research topic, the methods you have used, and the results you obtained.

Keywords: Resistor, Resistance,
Temperature

Preface

Thank you everybody.

Otaniemi, 9.3.2012

Albert Abelló Lozano

Contents

List of Tables

List of Figures

Definitions and abreviations

1 Introduction

The need of a new way to communicate between two points of the planet has been arising as a problem to be solved by our new media distributors. Traditional systems such Skype or video calls are not able to cope the needs of the new generations of developers and users that everyday require a more integrated way of communication with the World Wide Web (WWW).

Besides this, the amount of data being transferred during the last years and the prevision for the future allocates a new scenario where non-centralized systems such as P2P are required as data bandwidth grows and systems need to become more scalable. Nowadays, networks are still manly content-centric, meaning that data is provided from a source to a client in a triangle scheme, clients upload data to central servers and this data is transferred to the endpoint. This architecture has been provided since long time as reliable and scalable, but with the appearance of powerful applications and Video On Demand (VOD) it has been proven to become a real problem.

Those circumstances lead to a whole new world of real-time browser based applications which require also a totally new framework to be developed into. Starting from the online videoconferencing to real-time data applications, for this purpose few attempts where made in the past being highly reliable on specific hardware and custom-built no-compatible system. Taking these decisions made those proposals not be accessible by the massive amount of normal users that could not afford to adapt the requirements.

All the previous concepts are now possible thanks to the increase of performance related to the hardware components available in every average computer nowadays, this increase has helped to build more complex browsers that are able to perform many tasks not just related to web browsing. Having a browser handling OpenGL style of applications is now possible thank to the new HTML5 standard. Multimedia abilities have also been able to be reproduced on those browsers and handling webcam media as HTML is now a reality. Even dough there are still some issues to be considered before being able to freely communicate between two browsers: there is no common standardized protocol that allows developers to do this. WebRTC goal to approach this problem is to build a simple and standard solution for peer-to-peer browser communication [1] in the HTML5 environment.

Internet bandwidth has helped to take the decision to start integrating peer-to-peer solutions in browsed based applications, this is due the year-by-year increase of user bandwidth connectivity during the last 10 years. As before the latency was too high being unable to maintain real-time applications working resiliently. But recently the amount of users being able to transfer at high speed has been rapidly increasing as you can see in Figure 1, about 39% of users are now able to download at speeds greater than 4Mbps being this a very good average speed for media content [2].

Regarding the specs on the client side, recent surveys and statistics taken by the game manufacturer Steam [3] has proven that nowadays more than 61% of machines are carrying 1 to 4 gigabytes of RAM and nearly 90% of computers are handling 2 to

	Country	% Above 4 Mbps	QoQ Change	YoY Change
–	Global	39%	-2.8%	6.9%
1	South Korea	84%	-2.2%	28%
2	Switzerland	79%	2.3%	14%
3	Netherlands	79%	-5.0%	-2.0%
4	Japan	74%	2.4%	20%
5	Belgium	69%	-4.7%	-6.3%
6	Canada	69%	0.8%	8.2%
7	Hong Kong	68%	-5.2%	-4.7%
8	Latvia	67%	-1.7%	-2.8%
9	Denmark	66%	5.2%	5.0%
10	Czech Republic	64%	-6.4%	-10%
...				
13	United States	57%	-4.6%	7.9%

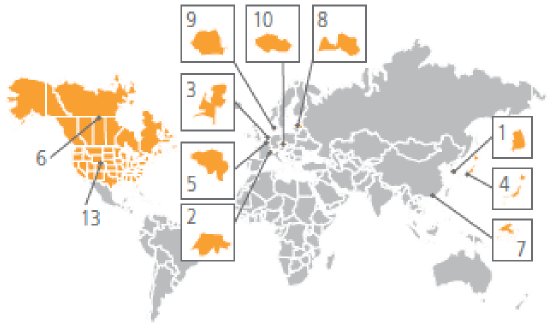

Figure 1: Broadband connectivity statistics about the speeds over 4Mbps around the globe.

4 core CPU with a 64 bit OS, being this environment optimistic for media enhanced applications which require high performance for video encoding and similar. Due to the layering needed to standardize a system like WebRTC running in top of an underlying application such as the browser that handles many processes is very important to rely on a powerful machine. Usually this was not able to be done in the past and one of the challenges has always been performance. They key to success is always to optimize the performance of a process so it does not affect the user experience, one of the challenges of WebRTC.

Traditionally, this concept of performance was approached by the usage of plug-ins or other separate software components which made the system run smoother by avoiding one layer of processing (browser) but being non-standard and not cross-compatible, one of the most import ant concepts when designing applications nowadays. Now the traditional approach has become outdated with the arrival of the new HTML5 where WebRTC is integrated as one of the new Application Programming Interfaces (API) available alongside other many different interesting capabilities.

1.1 Background

WebRTC API is included into the HyperText Markup Language version 5 (HTML5), this is the fifth version of the WWW language. This version includes different API's and JavaScript codes that help the developer to easily introduce new features into their already existing WWW applications. The initial HTML version (2.0) was published in November 1995 [4] with the only goal of delivering static content from the server to client browser. This language was used since 1990 even the standardization process ended in 1995, with the initial goal of serving data format that is portable to multiple platforms HTML became de de facto format for serving web information.

HTML is written in form of angle brackets (< >) named tags that contain different elements. Those tags are then interpreted by the browser to show the different data content served by the server. During the evolution of the WWW different new concepts where added to the HTML standard and new versions where published, things like JavaScript and Style Sheets made increased the flexibility and features of the WWW content enhancing the final user experience.

Due to the need of extending the features of the already existing HTML4 standard a new version was proposed in 2004 by the Mozilla Foundation and Opera Software [5]. This new proposition was focused in new developing technologies that could be backwards compatible with the already existing browsers, the idea didn't make a success and was tier apart until January 2008 when the first Public Working Draft was published by the Web HyperText Application Technology Working Group (WHATWG) in the W3C [6].

This proposal had a greater reliance in modularity in order to move forward faster, this meant that some specs that were included in the initial draft moved to different working groups in the W3C. Those technologies defined in HTML5 are now in separate specifications, one of them being WebRTC. WebRTC works as an integrated API within the browser that is accessible by using JavaScript and is used in conjunction with the Document Object Model (DOM) interfaces. Some of the APIs that have been developed are not part of the HTML5 W3C specification but are included into the WHATWG HTML specification.

1.2 Contribution

Investigate how WebRTC performs in a real environment trying to evaluate the best way to set multiple peer connections that are able to transfer media and data in different network topologies. Measure the performance of WebRTC in a real environment trying to identify bottlenecks related to encoding/decoding, media establishment or connection maintenance. All this should be able to be performed in real-time over a browser by using the already existing WebRTC API.

Using metrics related to RTT, latency, packet loss and bandwidth usage we expect to understand the way WebRTC performs when handling multiple connections.

1.3 Goals

WebRTC uses and adapts some existing technologies for real-time communication. This thesis will focus in studying how:

- WebRTC performs considering different topologies using video acquired by the API itself form the Webcam and encoded using different codec types provided by the standard.

- Usage of WebRTC to build a real application that can be used by final users proving that the API is ready to be deployed and is a good approach to the developer needs when building real-time applications over the web. This will be done in conjunction with other new APIs and technologies introduced with HTML5.

The final conclusion will cover an overall opinion and usage experience of WebRTC providing some valuable feedback for the needs and requirements for further modifications of the API.

1.4 Structure

Not sure about here

2 Conclusion

The end.

References

[1] H. Alvestrand. Overview: Real Time Protocols for Brower-based applications. Technical report, IETF, 2012. Also available in http://tools.ietf.org/html/draft-ietf-rtcweb-overview-04.

[2] Akamai. The State of the Internet, 2ND Quarter, 2012 Report, 2012.

[3] Steam. Steam Hardware and Software Survey, October 2012. Also available in http://store.steampowered.com/hwsurvey.

[4] T. Berners-Lee. HyperText Markup Language 2.0. Technical report, IETF, November 1995. Also available in http://tools.ietf.org/html/rfc1866.

[5] Mozilla Foundation and Opera Software. Position paper for the W3C workshop on web applications and compound documents. Technical report, W3C, 2004. Also available in http://www.w3.org/2004/04/webapps-cdf-ws/papers/opera.html.

[6] Ian Hickson and David Hyatt. HTML5. Technical report, W3C, January 2008. Also available in http://www.w3.org/TR/2008/WD-html5-20080122/.