International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 3, Juni 2020, pp. 2773~2778
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i3.pp2773-2778

Design and implement a new mechanism for audio, video and screen recording based on WebRTC technology

Naktal Edan, Sanabil A. Mahmood
College of Computers Sciences and Mathematics, Mosul University, Iraq

Article Info

Article history:

Received Mar 27, 2019
Revised Nov 27, 2019
Accepted Dec 12, 2019

Keywords:

4 Generation (4G)
Internet
Quality of experience (QoE)
Web real-time communication (WebRTC)

ABSTRACT

Many years ago, Flash was essential in browsers to interact with the user media devices, such as a microphone and camera. Today, Web Real-Time Communication (WebRTC) technology has come to substitute the flash, so browsers do not need the flash to access media devices or establish their communication. However, WebRTC standards do not express precisely how browsers can record audios, videos or screen instead of describing getUserMedia API that enables a browser to access microphone and camera. The prime objective of this research is to create a new WebRTC recording mechanism to record audios, videos, and screen using Google Chrome, Firefox, and Opera. This experiment applied through Ethernet and Wireless of the Internet and 4G networks. Also, the recording mechanism of this research was obtained based on JavaScript Library for audio, video, screen (2D and 3D animation) recording. Besides, different audio and video codecs in Chrome, Firefox and Opera were utilised, such as VP8, VP9, and H264 for video, and Opus codec for audio. Not only but also, various bitrates (100 bytes bps, 1 Kbps, 100 Kbps, 1 MB bps, and 1 GB bps), different resolutions (1080p, 720p, 480p, and HD (3840* 2160)), and various frame-rates (fps) 5, 15, 24, 30 and 60 were considered and tested. Besides, an evaluation of recording mechanism, Quality of Experience (QoE) through actual users, resources, such as CPU performance was also done. In this paper, a novel implementation was accomplished over different networks, different browsers, various audio and video codecs, many peers, opening one or multi browsers at the same time, keep the streaming active as much as the user needs, save the record, using only audio and/or video recording as conferencing with full screen, etc.

Corresponding Author:

Naktal Edan,
College of Computers Sciences and Mathematics,
Mosul University,
Mosul, Iraq.
Email: edannaktal@gmail.com, naktal.edan@uomosul.edu.iq

1. INTRODUCTION

Web Real-Time Communication (WebRTC) was developed by The Internet Engineering Task Force (IETF) and World Wide Web Consortium (W3C) [1-5]. WebRTC is a new standard and a collection of libraries [6] that supports interactive communications of video and data [7-10]. Additionally, it provides many benefits such as no fees, no license, no plug-ins, no installation, and so on [11-13]. In [14], emphasised that recording has changed the education route for delivering and consuming, Besides, many Application Programming Interface (APIs) have provided by WebRTC to be used in the screen capture and the media recording, also as clarified in the corresponding W3C drafts that have implemented in new browsers [15]. On the other hand, WebRTC lacks high-end videoconferencing types like a recording of a session [16]. Furthermore, [14] expounded that WebRTC standards have not realised what is happening on desktop

capture (screen) in addition to the recording of all media content. Indeed, WebRTC technology investigates reasons regarding all surfaces in demand to apply web application for streaming and recording supported from microphone, camera, and screen. (a) The essential objectives of this research are to design and test a WebRTC recording mechanism for audios, videos, and screen, (b) apply the created mechanism via Ethernet and Wireless of the Internet and 4G networks, (c) utilised various audio and video codecs, such as VP8, VP9, H264 and Opus audio codec, (d) different bitrates, such as (1 Kbps, 100 Kbps, 1 Mbps, and 1 GB) were used, (e) assortment of resolutions such as (1080p, 720p, 480p, and HD (3840* 2160)) were considered, (f) various frame-rates such as (5, 15, 24, 30 and 60) were tested, and (g) an evaluation of recording mechanism, Quality of Experience (QoE) through actual users and CPU performance was also done. Consequently, a novel implementation was accomplished over different networks, different browsers, various audio and video codecs, different peers, opening one or multi browsers at the same time, keep the streaming eactive as much as the user needs, save the record, and using only audio and/or video recording as conferencing with full screen. The organization of this project is as describes; Section 2 talks about the survey and WebRTC recording related work. In section 3, a preview of the methodology of the paper is explained with implementation and analysis. Section 4 relates to the evaluation. Finally, Section 5 is the conclusion and future work.

2. RELATED WORK

In [14, 16-19], explained that media capture, media screen, and media stream recording have considered as main WebRTC issues; especially recording API has not implemented yet. Additionally, recording in the browser may be unauthorized as long the server is in the media path. However, in [20] claimed that an application of screen recording with WebRTC on android was designed using getUserMedia API, but practically the author has not proved or presented the work. What is more, in [21] expected that using MediaRecorder API can support recording in WebRTC. On the other hand, in [22] illustrated that session recording is a significant challenge while stream mechanism has not provided by WebRTC standard to gather the information and store them. Accordingly, in [18, 23] expounded that WebRTC needs some solutions such as recording functionality to allow involvement of devices in limited network environments, and offer a WebRTC conferencing prototype that assists recording of conversations. Besides, in [11] confirmed that during the test, a screen recorder is necessary to record the data, such as video and audio, obtain user feedback and evaluate the quality.

3. METHODOLOGY, IMPLEMENTATION, AND ANALYSIS

3.1. Methodology

Methodology for designing and testing this application, different Libraries for audio, video, screen, canvas (2^{nd} and 3^{rd} animation) was used for the implementation and recording to designing and test this application. Also, JavaScript language, a task manager to evaluate a CPU performance, access Point-NetCommWireless to provide 4G, and Cameras and Microphones were used. Furthermore, Google Chrome, Opera, and Firefox were utilised as a client-side. Additionally, one computer connected through (Ethernet and Wireless) of the Internet and 4G networks.

3.2. Implementation

In this research, a new mechanism for video, audio, and screen recording has been designed and implemented based on RecordRTC library, MediaRecorder API and WebRTC JavaScript code for audios, videos, screen and canvas (2^{nd} +3^{rd} animation) recording. This application has divided into the following parts:

- Setup the main browser (Index HTML).
- Utilised RecordRTC API.
- Utilised node.js server for localhost server.

3.2.1. Setting up a browser web page

The first step will need to grant using the getUserMedia API in order to access the camera and microphone. So, RecordRTC will be able to start the video recording. RecordRTC and adapter scripts were needed to provide cross-browser for supporting getUserMedia and other browsers APIs. Moreover, different JavaScript methods were written in order to build this application as showing: (a) Start recording, Stop recording, and Pause recording, (b) Initialise recording, (c) Audio and video recording in different format such as Gif and Webm, (d) Auto-stop the recording after 5 minutes (to reset all the recorded data), (e) Save the recording into disk, (f) RecordRTC-Configuration.js, (g) GetRecorderType.js, (h) MRecordRTC.js

(to bring multiple records in a single place), (i) MediaStreamRecorder.js, (j) StereoAudioRecorder.js, (k) CanvasRecorder.js, (l) DiskStorage.js, (m) MultiStreamRecorder (to record multiple videos in single container), (n) Add extra media-streams (to existing recordings), (o) RecordRTC.promises.js, (p) WebAssemblyRecorder.js, etc.

It sets a video width to offer more clearance as width = 640 and high 480, as shown in Figure 1. When an initiator opened the browser, it will present audio and video "MediaStream," which can be obtained via "navigator.getUserMedia" method to capture screen. Once access to the camera and microphone; a media will start streaming the video and/or audio and display, and then saving them on disk. Figure 2 shows the pseudocode of this experiment. To leave the room or change the setup like resolution, bitrates, and codec, a user needs to close or reopen the web page. Also, can control the streaming of the camera or microphone, maximize the screen or pause anytime.

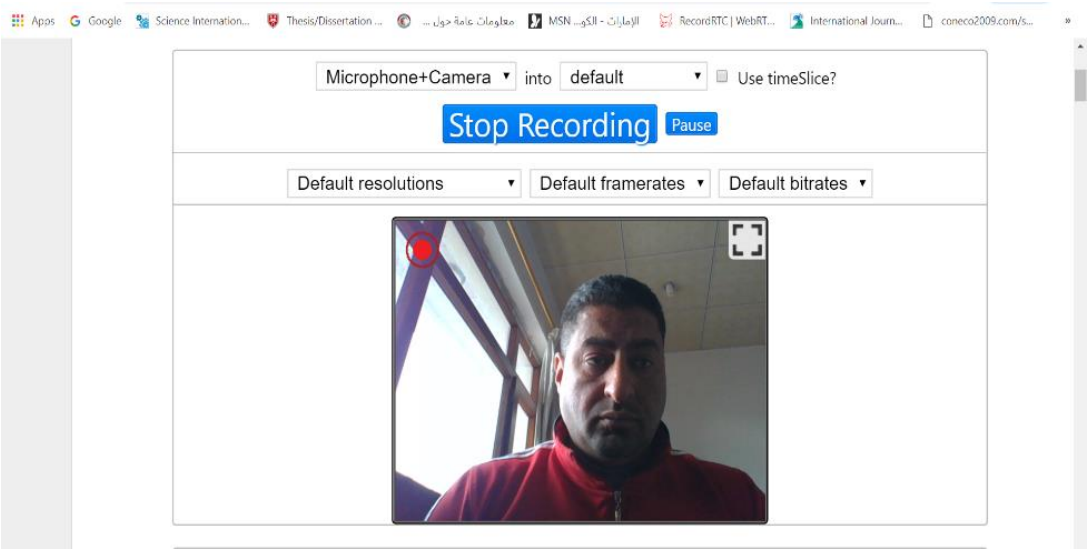

Figure1. The primary web page using firefox

```
1  SET RecordRTC API;
2  SET JavaScript Methods;
3  SET Node.js Server;
4  SET R1= Access Camera and Microphone;
5  SET ST= Start Streaming;
6  SET PS= Pause Streaming;
7  Get Media Stream;
8  SWITCH Start Recording;
9     CASE1: open a new page;
10          STEP1: internal access devices;
11                IF RI = yes;
12                     THEN open camera and microphone;
13                ELSE ask to access the internal devices;
14          STEP1: start streaming;
15                IF ST = yes;
16                     THEN stream media and save to disk;
17                ELSE ST= stop streaming and play recorded media;
18          STEP2: pause streaming;
19                IF PS = yes;
20                     THEN PS = stop streaming and recording;
21                ELSE PS = resume streaming;
22          STEP3: change the setting;
23                IF CR= yes;
24                     change resolution, frame rates, bitrate, codec, etc.;
25                ELSE leave it by default;
26    END
```

Figure 2. Implementation pseudocode

3.2.2. The created mechanism

This mechanism has been created and used based on RecordRTC API and JavaScript Methods. The RecordRTC library has been used to initialize and set up a new session for audios, videos, and screen using Google Chrome, Firefox, and Opera. Also, it has added many different JavaScript functions and methods for local and remote media streams.

3.3. Analysis

3.3.1. Signalling mechanism for saving media stream

This mechanism has been analyzed separately for ten users based to test the delay to get ready. This was implemented based on the network analysis inspecting element of Google Chrome, Firefox, and Opera, at the actual communication. The mean time was calculated, so it expands ***132*** milliseconds (ms) to be ready and consumes ***475*** (ms) to establish media streaming. This mechanism can set up, establish and send audios, videos, and screen simultaneously. The variation of delay between using Chrome, Firefox, and Opera was slightly different. In contrast, the quality of audios, videos and screen did not affect by the CPU load or the bandwidth consumption; especially the media streaming was between the internal devices, such as camera and microphone with the disk in the computer.

3.3.2. Quality of video conferencing

The quality of audios, videos and screen were done by individual test between ten users over the Internet and 4G networks. Therefore, the quality of audios, videos and screen were excellent. As a result, using the created mechanism for recording sounds, videos, and screens is efficient, as shown in Table 1.

Table 1. Quality of the audio, video, and screen betweenten peers over (LAN & WAN) of the internet and 4G networks

No.	Browsers	Operating System	Features	Codecs		Duration	Quality of audio	Quality of video	Quality of Screen
				Video	Audio				
1.	Google Chrome	Windows 10 and Ubuntu	Audio, Video, and Screen	VP8, VP9, H264	OPUS	1 - 5 minutes	Excellent	Excellent	Excellent
2.	Firefox	Windows 10 and Ubuntu	Audio, Video, and Screen	VP8, H264	OPUS	1 - 5 minutes	Excellent	Excellent	Excellent
3.	Opera	Windows 10 and Ubuntu	Audio, Video, and Screen	VP8, VP9, H264	OPUS	1 - 5 minutes	Excellent	Excellent	Excellent

3.3.3. Quality of experience (QoE)

In [24] mentioned that QoE is necessary and has been considered as a subjective matrix in media communication, as well as it has been adopted by ITU-T group [23, 25, 26]. Through the use of a questionnaire, users have taken part in this test to give their individual perspectives on the realized user experience as presented in Table 2. This application confirmed an excellent quality of audio, video and screen recording, in specific between ten peers via the Internet and 4G networks.

Table 2. QoE of ten users for communication via the internet and 4G networks

Questions	Very Bad Very annoying	Bad Annoying	Fair Slightly annoying	Good Perceptible but not annoying	Excellent Imperceptible
Rate the streaming using RecordRTC library			2	3	5
Rate the ease of using the application					10
Rate the quality of audio during the session				2	8
Rate the quality of the video during the session				1	9
Rate the quality of the screen during the session			1	1	8
Rate the resilience and flexibility of the connection			1	2	7
Does the RecordRTC library convince you to use WebRTC in the future				3	7

4. EVALUATION

In this research, it has been proved that the designed application can be utilised to support audio, video and screen recording among different browsers such as Google Chrome, Opera, and Firefox. This implementation is using a new WebRTC recording mechanism to set up, establish, stop and save media streaming over the Internet and 4G networks. Besides, it offers audio and/or video conferencing and keeps the media streaming productive, and controls self streams. Whereas, it has been created without using any external devices and commercial cloud/server. This experiment can be considered as the first one that achieved a WebRTC recording mechanism for audio, video and screen recording using the 4G network. However, it does not support Safari browser. There were not any significant issues with the performance of CPU and bandwidth consumption in audio or video streaming, while media streaming in this kind of data does not request a high processor for decoding, encoding, etc. The QoE ascertains that this testbed environment works correctly and that it is possible to use it to conduct further extensive experiments on user expertise in the future.

5. CONCLUSION AND FUTURE WORK

In this project, a new WebRTC recording mechanism via the Internet and 4G was established and tested in a real-time implementation. Furthermore, RecordRTC library was used to set up, create and end a media streaming for recording. This consequence is effective since it delivers visually demo over the different networks and browsers a real face-to-face communication. Moreover, it enhances communication & improves relationships and increase productivity between users and teams. Additionally, this experiment can be applied in various applications such as entertainment, lecturer between teachers and students, and meeting between doctors and technicians. In the future, there is an intention to extend this effort over extra scalable audio, video and screen recording using Safari and 5G in WebRTC.

REFERENCES

[1] B. Y. Julian. Jang-Jaccard, Surya. Nepal, Branko. Celler, "WebRTC-based video conferencing service for telehealth," *Computing*, vol. 98, no. 1–2, pp. 169–193, 2016.

[2] N. Edan, A. Al-Sherbaz, and S. Turner, "Design and implement a hybrid WebRTC signalling mechanism for unidirectional & bi-directional video conferencing," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 1, pp. 390-399 2018.

[3] N. M. Edan, A. Al-sherbaz, and S. Turner, "WebNSM: A Novel WebRTC Signalling Mechanism for One-to-Many Bi-directional Video Conferencing," in *Proceedings of 2018 SAI Computing Conference*, pp. 1–6, 2017.

[4] S. Perreault, "Traversal Using Relays around NAT (TURN) Extensions for TCP Allocations," USA, 2010.

[5] V. P. I. Baz Castillo, J. Millan Villegas, "The WebSocket Protocol as a Transport for the Session Initiation Protocol (SIP)," Spain, 2014.

[6] M. Phankokkruad and P. Jaturawat, "An Evaluation of Technical Study and Performance for Real-Time Face Detection Using Web Real-Time Communication," in *International Conference on Computer, Communication, and Control Technology (I4CT)*, no. I4, pp. 162–166, 2015.

[7] M. L. Giuliana. Carullo, Marco. Tambasco, Mario. Di Mauro, "A Performance Evaluation of WebRTC over LTE," in *12th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pp. 170–175, 2016.

[8] N. M. Edan, A. Al-sherbaz, and S. Turner, "Design and Evaluation of Browser-to-Browser Video Conferencing in aWebRTC," in *2017 Global Information Infrastructure and Networking Symposium (GIIS)*, pp. 4, 2017.

[9] S. Owesen-Lein, "Unified Communication and WebRTC," Norwegian University of Science and Technology, 2015.

[10] C. J. S. Nandakumar, "Annotated Example SDP for WebRTC: draft-IETF-rtcweb-SDP-09," USA, 2018.

[11] L. O. D. N. Eirik. Fosser, "Quality of Experience of WebRTC based video communication," Norwegian University of Science and Technology, 2016.

[12] N. M. Edan, A. Al-sherbaz, and S. Turner, "WebNSM : A Novel Scalable WebRTC Signalling Mechanism for Many-to-Many Video Conferencing," in *3rd IEEE International Conference on Collaboration and Internet Computing (CIC)*, vol. 2, pp. 1–7, 2017.

[13] T. F. Michael. Adeyeye, Member, Ishmeal. Makitla, "Determining the signalling overhead of two common WebRTC methods: JSON via XMLHttpRequest and SIP over WebSocket," in Africon, Pointe-Aux-PimentsConference, pp. 0–4, 2013.

[14] S. Skrødal, "SA8T2 Internal Deliverable Technology Scout : Stream and record lectures with WebRTC," 2016.

[15] N. PINIKAS, "A Webrtc Based Platform for Synchronous Online Collaboration and Screen Casting," Technological Educational Institute of Crete, 2016.

[16] M. Pasha, F. Shahzad, and A. Ahmad, "Analysis of challenges faced by WebRTC videoconferencing and a remedial architecture," I*nt. J. Comput. Sci. Inf. Secur.*, vol. 14, no. 10, pp. 698–705, 2016.

[17] B. Bos, E. Davies, L. Desmet, S. Farrell, M. Johns, and R. Wenning, "Strategic Research Roadmap for European Web Security," 2014.

[18] M. Walter, "WebRTC multipoint conferencing with recording using a Media Server," Stuttgart Media University, 2015.
[19] J. Rodriguez, Pedro. Cerviño Arriba, Javier. Trajkovska, Irena. Salvachua, "Advanced video conferencing based on webrtc," in *IADIS Multi Conference on Computer Science and Information Systems*, pp. 6, 2019.
[20] P. Kinlan, "Screen recording on Android with getUserMedia and WebRTC," 2016. [Online]. Available: https://medium.com/dev-channel/screen-recording-on-android-with-getusermedia-and-webrtc-c32ba9d29c28. [Accessed November 25, 2019].
[21] S. Penadés, "Record almost everything in the browser with MediaRecorder," 2016. [Online]. Available: https://hacks.mozilla.org/2016/04/record-almost-everything-in-the-browser-with-mediarecorder/. [Accessed Oct. 7, 2019].
[22] P. Rodríguez, J. Cerviño, I. Trajkovska, and J. Salvachúa, "Advanced Videoconferencing Services Based on WebRTC," in *IADIS International Conferences Web Based Communities and Social Media 2012 and Collaborative Technologies*, pp. 180–184, 2012.
[23] B. García, L. López-Fernández, F. Gortázar, and M. Gallego, "Practical evaluation of VMAF perceptual video quality for webRTC applications," *Electron.*, vol. 8, no. 8, pp. 1–15, 2019.
[24] R. C. Streijl, S. Winkler, and D. S. Hands, "Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives," *Multimed. Syst.*, vol. 22, no. 2, pp. 213–227, 2016.
[25] T. Hoßfeld, R. Schatz, M. Varela, and C. Timmerer, "Challenges of QoE management for cloud applications," *IEEE Commun. Mag.*, vol. 50, no. 4, pp. 28–36, 2012.
[26] K. L. and J. Z. H. Kim, "In-service Feedback QoE Framework," in *Third International Conference on Communication Theory, Reliability, and Quality of Service*, Athens/Glyfada, pp. 135–138, 2010.