WebRTC Error Handling

MS Mohamed Rifnas, DulanNimesh, RavinduBathiya, Dilan Madura Jayaneththi, DulminiWijesinghe and Mr. DhishanDhammearatchi

Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd

Abstract-Web Real-Time Communication (WebRTC) is an open-source real-time interactive audio, video communication framework and potentially useful standard that allows to incorporate features such as voice calling, video chatting, messages features and peer to peer (P2P) file sharing directly into the browser. New Firefox Hello video and chat client that lets user talk securely to anyone else using an up-to-date Firefox, Chrome, or Opera browser, without the need to downloaded any add-on, or configure any new settings. In WebRTC, Communication is extremely sensitive to affected by errors. Research paper discusses some of the mechanisms utilized in WebRTC to handle data missing, Virtual Private Network (VPN) bugs, Reporting errors and WebRTC traditional error in the communication path while WebRTC processing. In this case tried to solve the problems on the WebRTC communication. This Paper's reviews the techniques that have been developed for error detect, locate and correct in the WebRTC. More importantly, this system design can correct an error at a small modification of the basic structure.

Index Terms- Data Missing, Reporting Errors, VPN Bugs, Traditional Error

I. INTRODUCTION

Wired and wireless technologies have been developed with the wise spread of modes in communication. The vast usage of the communication devices such as computers, mobile phones, tablets the communication techniques will be increased. Most of the modern technologies are being emerged through traditional technologies. Web RTC is one of the fastest and quickest techniques that arise with above traditional technologies. IP communication was the prominent technology since last few decades. Internet Protocol (IP) communication has been prominent due to its wide spread all over the world. Communication Medias such as audio and video have been easy with the internet in modern period. Internet has been the foundation for WebRTC technique. Web RTC is a popular metric that is widely used in industries. It is able to interact with other browsers in real time which focus to a peer to peer connection. It is also an API definition develop by World Wide Web consortium. This enables facilities such as voice calling, video chat and peer to peer file sharing. This can be done easily without any plug-ins or third party software. It is a media engine available in all browsers launched by Google. WebRTC is mainly defined with Java Script Audio Video Interleaves (AVI), Standard Hypertext Markup Language 5 (HTML) tags and with some of the communication protocols [1].

Usage of this system is reduce due to network latency, packet loss and bandwidth. Due to above reasons errors are occur in WebRTC. This paper is present the errors that are occurring and the mechanisms which are used to handling this errors. VPN Bugs, Data packet missing, reporting errors, Queuing errors, traditional errors of WebRTC are further describe in the paper.

II. BACKGROUND AND RELATED WORK

WebRTC (real-time communication) web application developers the talent to write rich, real-time multimedia applications (video chat think) on the web, without plugins, downloads or installs. The tuning work WebRTC uses standard browser capabilities, JavaScript and HTML5 API to support real-time multimedia communication applications without using any browser plugins. Its purpose is to help build a strong platform RTC running on multiple web browsers across multiple platforms. Web applications advanced for WebRTC-enabled browsers can create a real-time communication with each other and with bequest network services. To access these applications, a subscriber needs to be connected to the Internet and the use of a device (such as a mobile phone, a laptop, a tablet or a desktop) gave with a WebRTC-enabled browser [3].

The problem of error control and concealment in video communication is becoming increasingly important because of the growing interest in video delivery over unreliable channels such as wireless networks and the Internet [1].

In video communications over error-prone channels, compressed video streams are extremely sensitive to bit errors. Often random and burst bit errors impede correct decoding of parts of a received bit stream. Video decoders normally utilize error concealment techniques to repair a damaged decoded frame, but the effectiveness of these error concealment schemes relies heavily on correctly locating errors in the bit stream [7].An important aspect in the development of software is to decide where to locate mechanisms for efficient error detection and recovery. This paper shows comparison between two methods for selecting locations for error detection mechanism. Especially for black-box modular software [6].

Memory and execution time reduce when proper placing EA's. The software's incorporate with error detection mechanisms (EDM's) and error recovery mechanisms (ERM's). Numerous algorithms and techniques have been proposed in logic circuits. (The D-algorithm, the PODEM algorithm FAN-algorithm) Propagation analysis in software has been described for debugging use in [8].

Error monitoring is very important process. It is very important source of information. By through error monitoring, error detection and correction will be much easier. The most likely neural generator of the ERN is anterior cingulated cortex (ACC), an area that in recent years has been implicated in another function related to the evaluation of performance, monitoring for competition (or conflict) during information. The Error-Related Negativity is focused on ERN, ERP. ERP components observed in at least three situations: following overt response errors in choice RT tasks, following late responses in deadline RT tasks, following feedback about response accuracy.

III. METHODOLOGY

The proposed consist four main part in order to accomplish handling the errors available in WebRTC they are Data packets missing, Error Repotting, VPN Bugs and Traditional Errors. Advance Mechanisms are each and every part of the proposed system to ensure error handling of WebRTC further main parts of this proposed system are describe below [11].

A. Data Missing

Web Real Time Communication (WebRTC) is an open-source project that enables web browsers with real-time audio and video communication. This part presents some of the underlying video processing aspects of WebRTC that enable reliable transmission of real time video over loss networks. It is well known that it is difficult to provide a high level user experience for associated real-time applications such as video call conferencing. These applications are limited by the time-varying nature of the network instruct (bandwidth, packet loss, network latency), and needs of low-latency real-time coding.

Different advent coexist [2] to handling packet missing during a multimedia transmission, such as packet re-transmissions based on forward error correction (FEC) [2], negative acknowledgment (NACK) and reference picture selection (RPS) [2]. These are often supplemented with codec error-resilience methods [2], such as intra-refresh and error concealment. For real-time applications with strict delay needs, a hybrid NACK/FEC scheme [2] can be used to achieve some balance of delay cost in the NACK method and repetition budget in the FEC method.

This part provide one set of amour control now a days using in WebRTC for how to handle packet miss. In actual, a flexible hybrid NACK/FEC method with temporal layers (TL) is advised as a helpful scheme to balance the quality of video, agility of and end-toned late. TL prepare to the temporal scalability feature in the VP8 codec made in WebRTC.

A.1 System Process Description

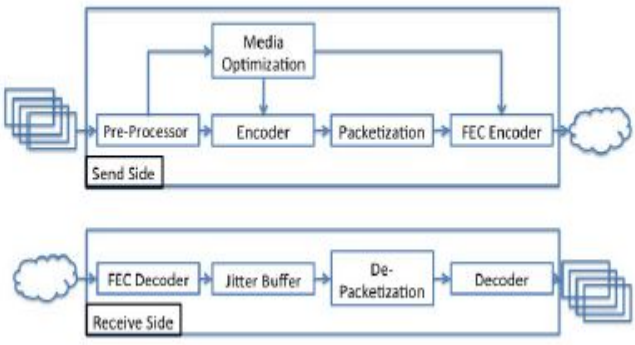

Figure 01: System Process
(Source:"https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC")

The figure 01 shows how the WebRTC video communication processing system work. Raw frames entering the send side are first preprocessed, and then encoded at a provide aim rate. After that, the frames are packetized, and when matching,
FEC encoder is affixed. The FEC is a XOR code based on RFC 5109 [2]. On the receiver side, packetized encoded data is impact by the FEC Decoder, tailed by the Jitter Buffer (JB). The latter compose encoded frames from the received packets and guess the video

jitter. Once a frame is done, it is sent to the decoder that outputs raw data (YUV format). The JB is also at fault for create a list of missing packets that are the basis for the re-transmission request.

Then the research save the per-frame statistics of the frame's capture duration and receive duration, The part model the jitter as build of two mechanisms, one casual and one relative to the size of the video frames [2]. And model it as mismatch dependent on the frame's size change. This method of guessing the jitter creates it possible to alter to difference in frame size and link capacity that often have a backlash on the video jitter. The jitter guess adapts to frames being late due to FEC decoding, but not due to re-transmissions.

The Media Optimization (MO) mechanism on the send side controls the flexible hybrid NACK/FEC. MO many times receives network records, which are updated with each and every incoming RTCP receiver report (approx. every second). These network statistics contain the expect bandwidth, fragment of packets lost and the Round Trip Time (RTT). A receive-side bandwidth estimator calculate the expect network traffic [2]. The MO also receives encoder statistics such as the incoming frame rate and the particular bitrates sent (video bitrate and FEC/NACK armor overhead rate). The important function of MO with regard to the hybrid NACK/FEC is to set the value of FEC armor, and update the encoder with the new source rate (expect bandwidth - estimated armor overhead).

A.2 System Behavior and Result

The system was calculated by using an offline mirroring tool, which simulates many network conditions in a handle environment. The mirroring tool acts like as a transport module in between the receiving client and the sending client, and comprise of a queue that ads a network transit late. A packet dropper is move after the queue, which can command packet misses drawn from a burst loss model using the Gilbert Elliot model [2]. In the following results, only done VP8 bit streams are provide to the decoder, in this situation the video is decoded without errors/packet miss, and the receiver is set to wait for all important packets. Video quality is therefore mainly impact by the smoothness of the playback and the expect bitrate.

A.3 Hybrid NACK/FEC

WebRTC uses an adjustable hybrid NACK/FEC method to archive a best trade-off in between temporal quality (smoothness of rendering), spatial video quality and end-to end delay. The adjustable aspects of this method assign to the dynamic setting of the FEC value at the sender side, and the play out delay at the receiver side.

The budget of the hybrid NACK/FEC method is the above penalty of the FEC, as displayed in Figure 2a. Apart from that, it has clear advantage over the NACK only method. Figure 2b displays that on average end-to-end delay is decreased when joining NACK and FEC since on average less time is spent waiting for re-transmissions, although the wait time for a single re-transmission is not changed. As shown in Figure 2c and d, the standard alteration of the render time delta is significantly decreased as well.

As mentioned in section 2, the FEC value (protection level) is observed in the MO based on the received network statistics records. In actual, the value of FEC is instruct on the RTT. Packets can be re-transmitted without substantial freezes when the RTT is low; therefore the value of FEC can be decreased, resulting in a smaller delay penalty. In large RTTs, the delay has a bigger effect on the temporal quality, and therefore the value of FEC should not be decreased. This is shown in Figure 2a, where the FEC overhead is decreased for RTT/2 < ~50 ms[2].

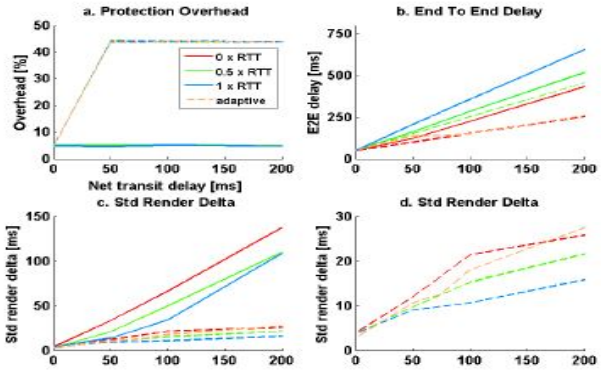

Figure 02: Hybrid NACK/FEC
(Source:"https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC")

Figure 02 – NACK and Hybrid NACK/FEC: for change values of d_{add}, as a function of network carrying delay. Solid lines refer to NACK, dashed lines refer to hybrid NACK/FEC. For a 5% packet miss rate, burst length of 1, at 500kbps.

The play out delay is controlled in the JB, and is used to trade-off the temporal quality (smoothness of rendering) with the end-to-end delay. The aim is to delay the playback in categorization to decrease the duration in which the video is frozen while waiting for a re-transmitted packet. However, when the RTT is too big, additional playback delay is less necessary, as one-way delays longer than 400 ms severely impair communication [2]. Therefore the supplementary playback delay should be accept depending on the fragment of beyond packet miss, u, and the guessed RTT. The supplementary playback delay can be computed as

$$d_{add} = min(max(K - RTT/2 - d_{jitter}, 0), RTT), \quad if\ u > U_{min},$$
$$d_{add} = 0, \quad otherwise.$$

Figure 03: Hybrid NACK/FEC Equation
(Source:"https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC")

Armor level set in the MO is such that the average armor overhead is ~20/25%. Figure 4c shows how the multi-frame in FEC can hit the aim armor overhead, while the 1-frame FEC overshoots and forward generates enough overhead, particularly for the least bit rate range (excess overhead decreases at higher bitrates, i.e., more packets/frame). The lower protection overhead for the multi frame case results in a higher Peak Signal-to-noise Ratio (PSNR)/quality [2].

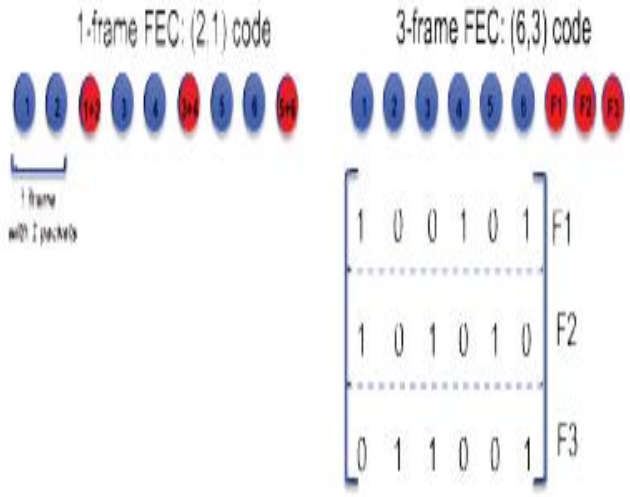

Figure 04: FEC Frame Code
(Source:"https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC")

Figure 04 - Multi-Frame FEC - Example of 2 packets per frame at 33% armor overhead (FEC packets obey the source packets they protect). The 1-frame FEC is the XOR of the two source packets in each and every frame, while the 3-frame FEC has the 6x3 generator matrix displayed above. The latter can recover more missed configurations, in actual any constant loss of size <=3 packets is completely retrievable with the (6, 3) code.

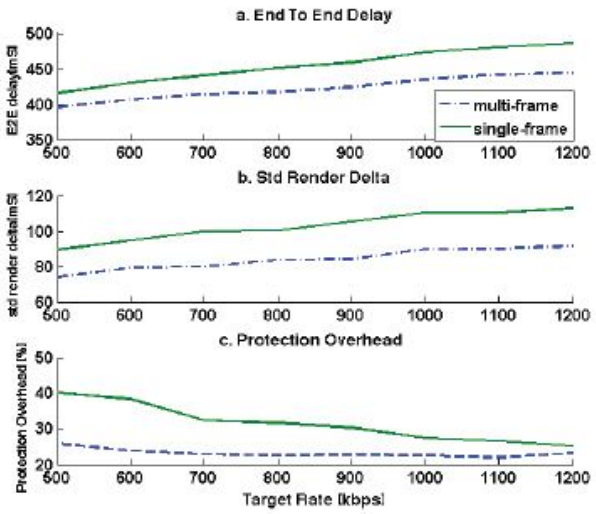

Figure 05: FEC Frame
(Source:"https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC")

Figure 05 Hybrid NACK/FEC :- One frame FEC vs. Multi-Frame FEC for a packet missing rate of 5%, burst length of 2, and RTT = 300ms; $d_{add} = 0$.

B. Error Reporting

When the functions are not performing well there is mechanism to show the errors to the users. This will helps to the users for better work of the system [6].

B.1 General Principles

When running Web RTC some failures can be identified. There is a mechanism to report those errors. Throw an exception is the one of the mechanism which is using for reporting errors that can be identified in a synchronous fashion. Those errors can be identified the compiling and after that error states are throw back via exceptions. Except the failures that can be identified in a synchronous fashion Invoke the error callback mechanism has used [6].

Error objects which are defined through java scripts, message and names, line numbers and the stacks can be thrown as exceptions. Line numbers and stacks are providing in some browsers only. Mozilla provide that facility [6].

B.1.1 createOffer( ) [7]

When the creation of the offer by peers the errors maybe occur and that errors throw as exceptions.
Exceptions that have used: INVALID_CALLBACK, INVALID_CONSTRAINTS, INVALID_STATE [11]

B.1.2 createAnswe ( ) [7]

When the creation of the answer by peers the errors can be detected and can be thrown as exceptions.
Exceptions that have used: INVALID_SDP, INCOMPATIBLE_CONSTRAINTS [11]

B.1.3 Calls after pc.close( ) [7]

After pc.close( ) method has executed some methods generate InvalidStateError (e.g. addStream( )). But all the methods are not performing the same scenario. The method like createAnswer( ), do not generate the InvalidStateError. Because of this behavior it is too difficult to identify the proper error and confuse the programmers. To avoid such problem any methods can be generated InvalidStateError except close( ) method.

B.1.4 Multiple calls to addStream( ) with same stream [7]

This problem is not considering in current development stage. By throwing ResourseInUse this problem can be solving.

B.1.5 Trying to send on a closed DataChannel [7]

When get a try to send a data packets on a closed DataChannel current mechanism is increment the bufferAmount and those data which has gathered to the buffer abort silently without mentioning in any functions. To avoid this silently abort mechanism can be thrown an InvalidStateError.

B.1.6 Create RTCTMFSender on non-audio track [7]

Current procedure to create RTCTMFSender on non-audio track is create it with. canInsertDTMF ===false. Generate an InvalidParameter exception is also good way to fulfill this task.

B.1.7 Call insertDTMF( ) when .canInsertDTMF is false [7]

In such a situation silently ignore the data. This problem will be able to solve by generate InvalidStateError.

B.1.8 Insert DTMF with bogus values (Guduru) [7]

Current procedure to solve this reporting error method is also generate InvalidParameter exception.

Unused Errors:

- Incompatible Constraints Error
- Incompatible Media Stream Track Error

B.1.9 Sdp Error

Though this error is define but never used, even though it is semi-referenced.
The good way to solve this is Edit set { Local,Remote } description section tomakeclar I is passed to error callbacks.

B.2 ICECanidate

In order to establish a connection to exchange information with peers it is essential to have some sort of serves in the middle. This is known as the signal channel. Through this channel peer who will be start of the connection send offer to the peer B. When peer B receive offer, peer can create an answer. Session description is the end point configuration of the WebRTC connection. By using session description protocol (SDP) the information's like kind of media, format, endpoint's ip address and port address are exchanged. Further network connection information's must have exchanged by peers. This process is known as ICE candidate.

B.2.1 addICECanidate( ) in wrong state [7]

To achieve the better performance of this function those data should be queued first. Should get InvalidSteError when get try to addICECandidte(in wrong state.

B.2.2 addICECandidate( ) bogus data [7]

If the situation is ambiguous, when candidate applied unsuccessfully,
The session description errors mention “It is busted” and provide line numbers. But it is better if provide the InvalidCandidate, InvalidMid and InvalidLineIndex with the accurate meanings.

C. VPN Bugs

Communication network in real time (WebRTC technology) is a potentially useful standard that allows browsers include features such as voice calls, video chat and P2P file sharing directly in the browser. A good example of this is the new Firefox Hello video chat client that lets you securely with others who use the latest browser Firefox, Chrome or Opera, without having to download plug-ins, or configure new settings. Unfortunately, VPN users, the WebRTC technology allows a Web site (or other technology services WebRTC) directly detect the actual IP address of your host if you use a proxy or VPN server [8].

Firefox and Chrome have implemented WebRTC, allowing Session Traversal Utilities for NAT (Network Address Translation (STUN)) server queries, the user returns to local and public IP addresses. The results of these requirements available in JavaScript, so you can now get the user JavaScript and local public IP address. In addition, these STUN requests outside the normal procedures of Extensible Markup Language (XML) HttpRequest, so they are not visible in the Developer Console or through plug-ins, such as GhosteryAdBlockPlus or clogging. This makes these types of tracks available online application, if an advertiser has set up a STUN server generic domain [9].

Opera browser, which uses the same code that powers Chrome WebKit is also affected by this problem, but Internet Explorer and Safari browser does not support WebRTC technology, are not. Update: The stock Android browser with new versions of the WebRTC technology, so it should be avoided [10].

D. Traditional Errors

Denial of Service attacks (DoS attacks) are shaped on the transfer of a huge total of messages (signaling traffic) to collapse Central Processing Unit (CPU), bandwidth or memory of the attacked system or device. The most common objective is to obtain login, password or just delay numbers, by the use of brute force attacks. This info could be used for fraud or illegal interruption, then this attack can also force a crash in the Session Initiation Protocol (SIP) proxy or soft switch [3].

When an attacker knows the extension of a valid user and the password or detects a publicly accessible port not protected, it is possible to make an illegal use of the telephony service. In this case the objective is commonly to make calls to international destinations or premium numbers where it is possible to get important revenues in a short time. A stolen WebRTC account with a Public Switch Telephone Network (PSTN) number associated can cause important legal problems beyond financial losses [4].

Another important thing is to correctly alert calls to users and this is not controlled by any standard. It is common to use for both events ring back and ringing tones respectively, simulating the behavior of any other physical telephone or softphone. An open issue for WebRTC is to notify calls in smartphones when the browser is in background as a push mechanism must be used [4].

Here is another thrilling view regarding web sockets, temporarily they open a Transmission Control Protocol (TCP) socket finished which scripts are able to direct any kind of traffic. If this traffic looks exactly the same as HTTP request and response, intermediate basics can possibly take the traffic as valid and, for example, include fake pages in the cache [4].

In the case of DoS and Denial of Service (DDoS) attacks, as some other service intended to be existing on Internet, WebRTC services essential be protected against them. It will receive signaling traffic over TCP so it must implement all the basic protection against attacks. Servers should be able to dynamically blacklisting IPs from where they are receiving attacks [5].

IV. Conclusion

The web has revolutionized communication, and WebRTC revolution promises to take this a step further. Free, open source project allows compatible web browsers to communicate in real time using simple JavaScript APIs. Therefore, by this one can conclude that WebRTC points to the evolution of real - time communication service by using simple JavaScript APIs. It can thereby improve the quality of communication, which will be delivered through browsers. WebRTC is unleashing the power of real-time communication to

any Web or application developer. An important what happens WebRTC is the way it is cooked in browsers. WebRTC is generally regarded as allowing communication between browser users, but is likely to also be used to create new user interfaces for websites. However WebRTC promises and establishes a vision of future communication technology existing today, to certain point. Ultimately, the effect of VoIP on the masses seems to be evolving and with some extra efforts may WebRTC become a great help in the field of communication. The use of this application is reduced due to network latency, packet loss and bandwidth. Due to the above reasons WebRTC errors follow. This document is to present the errors that are going on and the mechanisms used to handle this error. When functions are not performing well there is mechanism to show the errors to the users. Finally VPN errors, missing data packet, error information, queuing delay algorithms described in more detail in the document [1].

V. Future Work

Future research, although more effective error concealment even more emphasis approaches are necessary it must be placed in the system-level design and optimization in which the encoding algorithm, the transport protocol, and post-processing method should be designed jointly to minimize distortion due to the combined compression and transmission. In addition, optimal system should adapt its algorithm and source coding Transport Control mechanism for network conditions so that the best end-to-end service quality is achieved.

Best system should concealment assign redundancy between the source and transport layers encoder adaptively based on channel environment in order to optimize the video reconstructed quality for a given capacity Error-concealment decoder. This remains a difficult task for future research and standardization efforts.

Acknowledgment

Authors are very much thankful to Sri Lanka Institute of Information Technology for doing this research work in Computer Networking as well as all the authors whom team have referred in this research paper. Authors are also thankful to Mr. DhishanDhammearatchi for his constant support and encouragement for conducting research work in Computer Networks Design and Implementation.

References

[1](http://www.webrtc.org/;),
(http://code.google.com/p/webrtc/), [Accessed Date: 2016-07-07]

[2] Holmer, Stefan, Mikhal Shemer, and Marco Paniconi. (2013): "Handling Packet Loss In Webrtc". 2013 IEEE International Conference on Image Processing n. pag. (https://www.researchgate.net/publication/269031544_Handling_packet_loss_in_WebRTC) [Accessed date: Web. 15 Sept. 2016]

[3] Rahaman, Md. Habibur, (2015). "A Survey On Real-Time Communication For Web" (https://www.w3.org/2011/04/webrtc/wiki/images/0/0f/Webrtc-error.pdf) [Accessed Date: 2016-09-6]

[4] Jian, Cui. (2015). “Research And Implementation Of Webrtc Signaling Via Websocket-Based For Real-Time Multimedia Communications", (https://www.google.lk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwiXtZLfsJ7OAhUYS48KHRoYDMwQFggaMAA&url=http%3A%2F%2Fwww.atlantis-press.com%2Fphp%2Fdownload_paper.php%3Fid%3D25848184&usg=AFQjCNEzniXE2oZf0zR-fSuKG96q7uffpw&bvm=bv.128617741,d.c2I&cad=rja) [Accessed Date: 2016-09-15]

[5] Birajdar, Pooja. (2016). “Restricting Untrusted Browsers’ Communication In Web-RTC", (http://www.ijrter.com/papers/volume-2/issue-3/restricting-untrusted-browsers-communication-in-web-rtc.pdf), [Accessed Date: 2016-08-24]

[6] E. Rescorla, Web RTC Error Reporting, (2016), pp. 212. (https://www.w3.org/2011/04/webrtc/wiki/images/0/0f/Webrtc-error.pdf), [Accessed Date: 2016-09-04]

[7] A. Narayanan, "WebRTC: Error Handling", 2016. [Online].Available: (https://www.w3.org/2011/04/webrtc/wiki/images/d/d1/TPAC_2012_WebRTC_error.pdf.) [Accessed: 14- Sep 201s6].

[8]Rahaman, Md. Habibur, (2015). "A Survey On Real-Time Communication For Web", (http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=994913&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D994913), [Accessed Date: 2016-09-6]

[9] Jian, Cui. (2015). “Research And Implementation Of Webrtc Signaling Via Websocket-Based For Real-Time Multimedia Communications", (http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1687424&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D1687424), [Accessed Date: 2016-09-15]

[10] Birajdar, Pooja. (2016). “Restricting Untrusted Browsers’ Communication In Web-RTC",(http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=207595&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D207595),[Accessed Date: 2016-08-24]

[11] http://google.lk/

AUTHORS

First Author – MS. Mohamed Rifnas, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd, mrifnas345@gmail.com
Second Author –DulanNimesh, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd,
Third Author – RavinduBathiya, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd,
Fourth Author- Dilan Madura Jayaneththi, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd,
Fifth Author- DulminiWijesinghe, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd,
Sixth Author- DhishanDhammearatchi, Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd,

Correspondence Author – MS. Mohamed Rifnas,Faculty of Computing, Sri Lanka Institute of Information Technology (PVT) ltd