WebRTC role in real-time communication and video conferencing

George Suciu
R&D Department
BEIA Consult International
Bucharest, Romania
george@beia.ro

Stefan Stefanescu
R&D Department
BEIA Consult International
Bucharest, Romania
stefan.stefanescu@beia.ro

Cristian Beceanu
R&D Department
BEIA Consult International
Bucharest, Romania
cristian.beceanu@beia.ro

Marian Ceaparu
R&D Department
BEIA Consult International
Bucharest, Romania
marian.ceaparu@beia.ro

***Abstract*— Real-time communication (RTC) is a new standard and industry-wide effort that expand the web browsing model, allowing access to information in areas like social media, chat, video conferencing, and television over the internet, and unified communication. These systems users can view, record, remark, or edit video and audio content flows using time-critical cloud infrastructures that enforce the quality of services. However, there are many proprietary protocols and codecs available that are not easily interoperable and scalable to implement multipoint video-conference systems. WebRTC (Web Real-Time Communication) is a State-of-the-Art open technology that makes real-time communication capabilities in audio, video, and data transmission possible in real-time communication through web browsers using JavaScript APIs (Application Programming Interfaces) without plug-ins. This paper aims to introduce the P2P video conferencing system based on Web Real-Time Communication (WebRTC). In this paper, we have proposed a web-based peer-to-peer real-time communication system using the Mozilla Firefox together with the ScaleDrone service that enables users to communicate with high-speed data transmission over the communication channel using WebRTC technology, HTML5 and use Node.js server address. Our experiments show that WebRTC is a capable building block for scalable live video conferencing within a web browser.**

We also study a thermal camera for the videoconference system to identify the temperature body results regarding the COVID 19 crisis.

***Index Terms*—real-time, WebRTC, Node.js server, HTML5, JavaScript API**

I. INTRODUCTION

The open-source WebRTC enabled users of these systems to view video content or record, comment on stream it to achieve real-time communication between web browsers. WebRTC is a real-time communication technology that has integrated standards of API (Application Programming Interface) with real-time multimedia transfer such as voice and video (including codes) available to a web browser without traditional plug-in components using JavaScript code [1][3].

Recently, there was an increase in the new platform implementation of real-time communications services: browser embedded application or "web application." Among these applications, WebRTC has received significant interest since a lot of new versions of inherently supports this API common browsers, namely Google Chrome and Mozilla Firefox. WebRTC, which relies on HTML5 web communication, holds Peer Connection, Media Stream, and Data Channels components API that can be combined to create P2P direct media communication between peers. The current version of the WebRTC API was designed only to support browser-to-browser communication. WebRTC for "Multi-browser" communication is not inherently recommended, especially for conference models that spread the media load over participating peers/browsers [5].

WebRTC approaches more like a WebSocket, but WebSocket opens a pipe of connection with a server instead of another peer. In most cases, these technologies are used together for signaling purposes. In chat applications, for example, WebSocket clients first send messages to the server, and the server sends the messages to the recipients.

WebRTC promises to provide secured direct P2P communication between users and free of plug-ins. WebRTC assures a simplified, flexible, and cost-effective means of real-time communication for users without dependence on service providers. A critical challenge with plug-ins such as Flash, Silverlight, and Shockwave is the need for downloads each time a connection is to be established. Plug-ins can be problematic during execution; they increase bandwidth, latency, execution time, and speed [12].

Recent researches have made some exciting progress in WebRTC. Sodhoro Ali Hassan and Giancarlo Fortino

978-1-7281-6728-2/20/$31.00 ©2020 IEEE

implemented a battery recovery system based on wireless video transmission using a battery management method with the delayed adaptation of different components in the communication system [15]. Ali Hassan Sodhro et al.[16] developed a dynamic TPC based energy algorithm for human vital sign signal transmission in WBANs. Their algorithm saves a reasonable amount of energy with high RSSI stability, but it is somewhat complex and consumes more power than the proposed EEA.

In this paper, we propose a solution to a video conference system using:

- Scaledrone, which is a push messaging service, where you create a channel;
- WebRTC, for the website server to send the browser IDs to the visitor. This ID is unique, and browsers can then connect to a specific peer browser.;
- HTML for programing, with the core of the system a JavaScript API.

Section II of the paper presents the related work, describing the technological architecture of the WebRTC and the main protocols used. Then section III offeres the functionality of a TURN server using ICE (Interactive Connectivity Establishment) protocols for data transport. Section IV illustrates the experimental results of the video conference application designed using Scaledrone, which is a push messaging service and the Mozilla Firefox browser. Finally, in section V, the results are concluded.

II. Related Work

Before the existence of WebRTC, there were already many video conferencing systems available on the market. The most popular example is Skype. Microsoft is the company that owns Skype. Skype uses a proprietary protocol for the transmission of multimedia streams, plus it requires the installation of a mobile application or desktop to access services such as phone calls, messages, and video conferences. Still, it is not possible to integrate a phone call with an active video conference [16].

The WebRTC architecture provides end-to-end encrypted P2P communication with audio-visual content and data being transmitted directly. The implementation is realized to bypass intermediary hardware servers and eliminate security challenges like hijackers. This particular feature makes the difference between WebRTC and other RTCs such as Skype.

Skype is proprietary and lacks the direct P2P ability, as well as a credible security feature found in WebRTC.

Peer-to-peer also enables user's data to be encrypted, safe, and cannot be compromised. A peer-to-peer connection can be credible because it is able to bypass all the problems associated with plug-ins. Factors such as latency, bandwidth, and memory utilization as well as support for anonymity are supported in WebRTC.[13]

2.1. Architecture of WebRTC

Figure 1. shows the architecture of the WebRTC. In general, WebRTC consists of three parts:

- the API layer for Web developers;
- the API layer for browser developers;
- the custom service layer for browser developers.

The WebRTC (Fig. 1) contains a Voice Engine, Video Engine, and tools for Transport and communication. Web browsers and other native applications can access the framework through its C++ API. Web applications cannot access this low-level API for security and interoperability reasons, so web browsers need to provide another way for developers to use it. The standard way of doing this is through a JavaScript API. Web applications can use standardized JavaScript API to access the functionality of WebRTC [2][4].

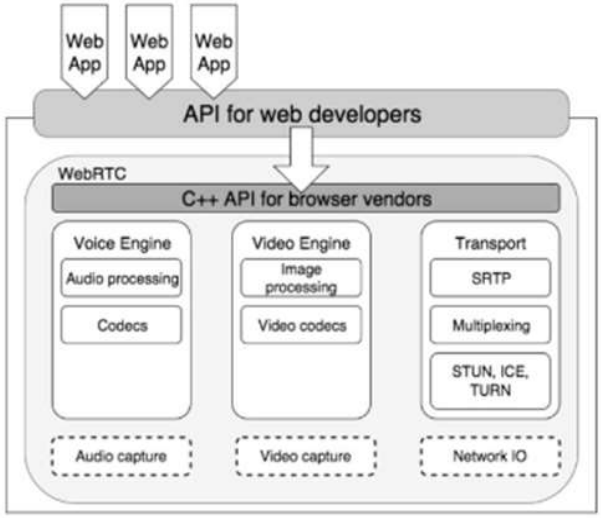

Fig. 1. The technology structure of WebRTC

Voice Engine is a function mainly responsible for audio processing. The features include audio decoding and sound processing.

Video Engine mainly deals with video codec and image processing. In video coding and decoding, WebRTC currently uses VP8 technology, which enables WebRTC to provide higher quality video images in a lower meta-rate environment. In the image processing section, WebRTC mainly includes functions such as jitter buffering and image enhancement to reduce the noise of images captured by the camera.

The transport function is mainly responsible for encrypting the collected audio and video, firewall penetration and transmission, and transmitting using SRTP (Secure Real-time Transport Protocol) protocol to ensure the security of the message when it is sent.[14]

The most common WebRTC Trapezoid model (see Figure 2), both browsers are running on a web application, downloaded from a different Web Server (or as usually in the field one shared server). A Peer Connection configures the path to flow directly between browsers without any interventions from servers. Signaling goes through HTTP or WebSockets, via Web Servers that can modify, translate, or manage signals as required. It is to be taken into account nothing that the signaling between server and browser is not standardized in WebRTC because it is considered to be part of the application. The two web servers can communicate by using a standard signaling protocol, such as Session initiation protocols (SIP) or Jingle [XEP-0166]. Otherwise, a proprietary signaling protocol can be used [7].

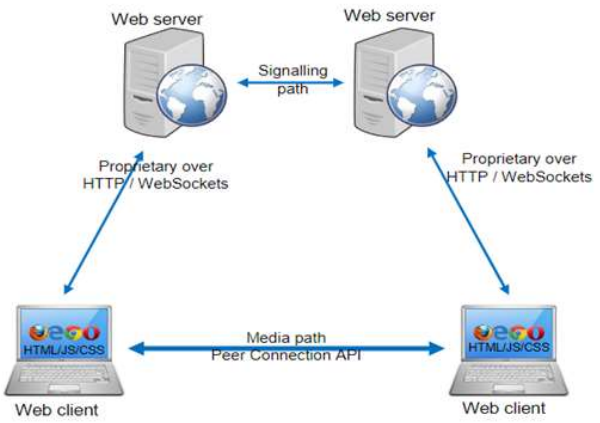

Fig. 2. The WebRTC Trapezoid

A WebRTC web application uses standard WebRTC APIs to allow it to exploit and control browser features in real-time properly. It has to do more things:

- get streaming audio, video, or other data;
- get information about the network, such as IP addresses and ports, and change this with other WebRTC clients (known as partners) to allow a connection, even though NAT and firewalls;
- coordinate signaling communication to report errors and initiate or close sessions;
- change information about media and client capacities, such as resolution and codecs;
- communicate streaming audio, video, or data [11].

To acquire and communicate streaming data, WebRTC is implementing the following APIs:

- MediaStream: Get access to data feeds, such as the user's camera and the microphone;
- RTCPeerConnection: audio or video calls with encryption and bandwidth management features;
- RTCDataChannel: peer-to-peer communication of generic data.

2.2. Protocols regarding WebRTC

To ensure a standard level of interoperability between different real-time browser implementations, Internet Engineering Task Force (IETF) works to select a minimum of audio and video codecs. Opus and G.711 the mandatory audio codecs to be implemented [8], and VP8 and H.264 Constrained Baseline as video codecs [9].

The API is being designed around the three main concepts: PeerConnection, MediaStream, and DataChannel.

The PeerConnection mechanism uses the Interactive Connectivity Establishment (ICE) protocol together with the Session Traversal Utilities for NAT (STUN), and Traversal Using Relays around NAT (TURN) servers to let User Datagram Protocol(UDP)-based media streams traverse NAT boxes and firewalls. ICE allows the browsers to discover enough information topology of the network where they are deployed, find the best exploitable communication path. Using ICE also provides a security measure because it prevents web pages and unencrypted applications from sending data to hosts that they do not expect to receive [6].

Data channels are created between peers using the RTCPeerConnection and an underlying transport built as Stream Control Transmission Protocol (SCTP) over Datagram Transport Layer Security (DTLS) over UDP.

This data transport is realized using SCTP

with some extensions. SCTP has native support for message transport and multiple flows with prioritization. SCTP does not impose any size limitation on messages. However, SCTP implementations have only limited send and receive buffers and, therefore, also provide non-atomic ways of sending and receiving to allow the transfer of messages larger than these buffers. To ensure confidentiality and authentication of SCTP packets, they are sent protected by a DTLS association. This DTLS association is run over a lower-layer transport flow provided by ICE, commonly UDP.

Real-time communication is a critical activity regarding the time that may result in intermittent packet losses during video streaming. The WebRTC audio and video codecs have surpassed this challenge by implementing various logic to recover from packet losses or delays. And at the same time, it considers timelines and low latency in data transmission as significant factors. WebRTC takes into account these factors more important than the reliability of data. That is the main reason why UDP protocol is the preferred over Transmission Control Protocol(TCP) for delivering real-time data. TCP provides a reliable and ordered stream of data. For instance, if an intermediate packet is lost, then TCP will buffer all the packets after it, wait for retransmission, and then delivers the stream to recover. At the same time, UDP offers no guarantee of message delivery or order of birth, no acknowledgments, retransmissions, or timeouts, no packet sequence numbers, no head-of-line blocking, no connection state tracking, establishment, or teardown state machines, congestion control, built-in client or network feedback mechanisms. As a result, UDP offers no reliability promise. UDP transport protocol, therefore, delivers each packet to the target application [12].

STUN provides the requesting endpoint of the public IP address. STUN is a relatively easy process because once STUN provides an IP address accessible to the public for the requester, it is no longer involved in the conversation (see Figure 3).

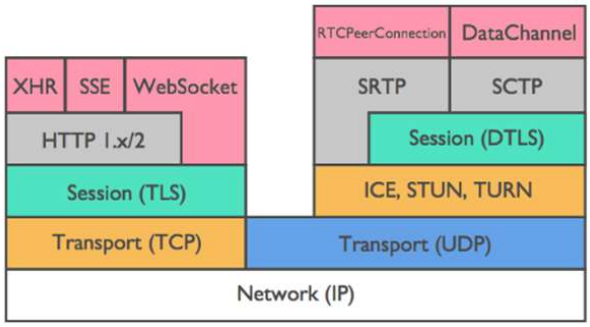

Fig. 3. The WebRTC Protocol Stack

In the case when an endpoint is behind a NAT, it only sees the local IP address. The other endpoints in the call could not use

this local IP address to connect to the endpoint, as it might be a private address or the firewall does not allow access. In such cases, this endpoint may require a STUN server to provide the public IP address. The participants then use the ICE procedures and try to establish a connection using the public IP address, and if the connection is configured successfully, the media stream is transmitted directly between the users without any active intermediary. For all practical purposes, STUN is down, waiting for the next query [10].

In some implementations of the NAT, the port will be translated to another port, along with the IP address to which it is attached. This situation is called "symmetric NAT." The public IP address of the STUN process is not enough to establish the connection here because the port would also require the translation; that's why a TURN server becomes essential [10].

III. Thermal Camera And NoIR Camera USing Raspbery PI For Video Conference

3.1. Thermal Camera

Thermal cameras, together with deep neural networks, are a much more robust strategy actually to detect the presence of people. In contrast with motion sensors, they will identify people even if they aren't moving. And, contrary, optical cameras can detect bodies by measuring the heat that they radiate in the shape of infrared radiation. And, contrary to optical cameras, they detect bodies by measuring the temperature that they emit in the shape of infrared radiation, and are therefore much better— their sensitivity doesn't depend on lighting conditions, of the position of the target, or the color.

Thermal imaging is more likely a non-destructive, non-contact, and rapid system. It reports temperature through measuring infrared radiation emanated by an object. Automated thermal imaging system involves thermal camera equipped with infrared detectors, signal processing units, and image acquisition systems supported by the computer. It is elaborated in wide domain applications. In medical applications, Thermography is an imaging technique that is used to measure the temperature distribution in organs as well as tissues. The visual display of this temperature distribution is called a thermogram. Thermography can be used in several conditions as a diagnostic tool for planning the treatment and evaluate the effects of treatment. Thermography if used with other imaging modality, play a vital role in the conformation of many diseases like COVID-19. All objects with a temperature above 0K emits electromagnetic radiation, which is known as infrared radiation or thermal radiation, which lies within a range of 0.75–1000 micrometers.

AMG8833 Camera (thermal infrared camera) is a non-contact device that detects infrared energy (heat) and converts it into an electronic signal that is processed to produce a thermal image on a video monitor. The signal can also be used to perform temperature calculations.

AMG8833 camera (thermal infrared camera) is a non-contact device that detects infrared energy (heat) and then converts it into an electronic signal that is processed to produce a thermal image on a video monitor. The message can also be used to perform temperature calculations.

Fig. 4. AMG8833 thermal IR sensors from Panasonic

The AMG8833 is defining the next generation of 8x8 thermal IR sensors from Panasonic and offers higher performance than it's predecessor, the AMG8831. The sensor supports I2C and has a configurable interrupt pin that can activate when any individual pixel goes above or below a threshold that you set. The sensor is from Panasonic and has an 8x8 array of IR thermal sensors.

3.2. NoIR Camera

The Raspberry Pi NoIR Camera Module is a custom designed add-on for Raspberry Pi that does haven't an 'IR cut filter' installed. Like the regular Pi camera, it attaches to Raspberry Pi by way of one of the two small sockets on the board upper surface. This interface uses the dedicated CSI interface, which was designed especially for interfacing with cameras.

Fig. 5.NoIR camera module for Raspberry Pi

IV. Experimental Results

This section describes the video conference system, by using Scaledrone, which is a push messaging service and an easy way to add real-time capabilities to your web or mobile app.

Scaledrone uses WebSockets when it is possible and goes back to technologies such as XHR streaming, JSONP polling, and XMLHttpRequest (XHR) polling when needed.

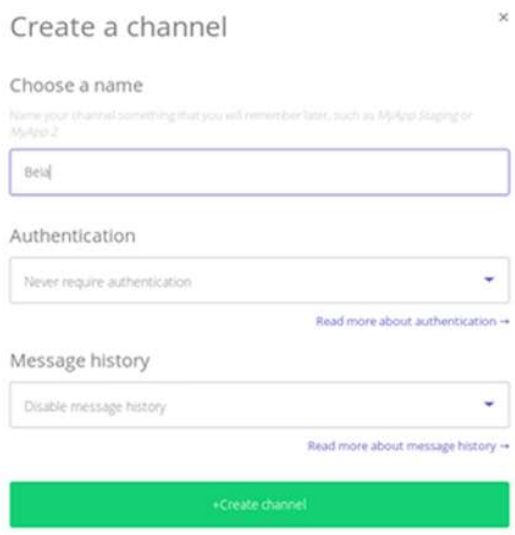

Fig. 6. Creation of a new channel in Scaledrone

Fig. 7. Unique ID that will be used in the app

In Scaledrone is created a new instance (channel), and in your channel's page, from the dashboard, you will find a unique ID that will be used in the app.

Real-time video conferencing implementation in Scaledrone is shown in Figure 7 and Figure 8. This experimental conference was established using an IP address: 10.0.8.173. For the video call we need to have the latest version of Mozilla Firefox (see Figure 9).

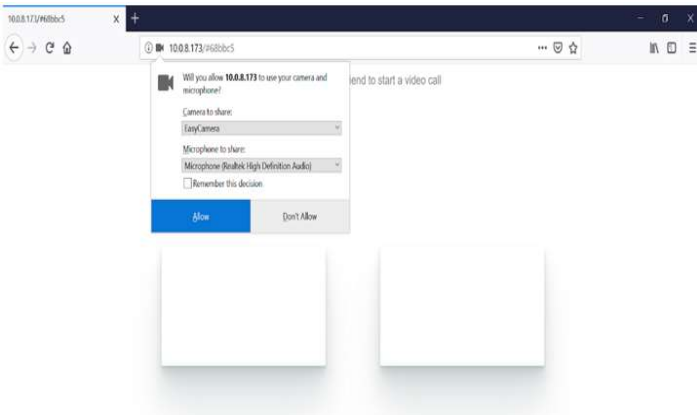

Fig. 8. Automatic code generation

The automatically generated code is then shared with the participants; the connection between participants is established (see Figure 10).

The interface shows a video conferencing communication between two users. The users are engaged in real-time interactions. It is a direct connection between the users' browsers devoid of any conventional DNS server connection between the users. The users' browsers did not need the support of any third-party plug-ins or downloaded software such as flash for the video to play on both browsers. The connection is possible because the getUserMedia() method establishes access to the cameras and microphones. Once the users enable the video conferencing button, WebRTC mandate a request for permission to use media devices. The users can then take any of the options to "allow" or "block" the request. Taking the "allow" option means that the system will have access to the users' camera and microphone for real-time interactions.

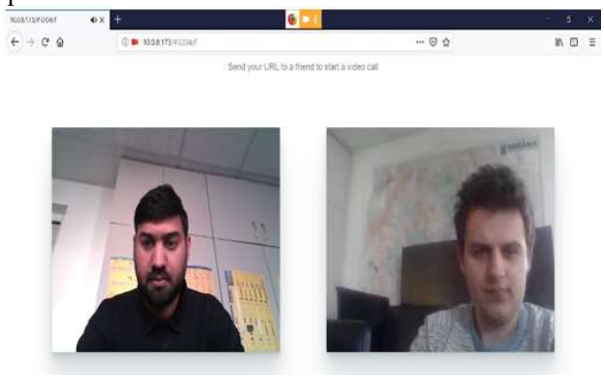

Fig. 9. Multi-party video conferencing: Connection established

For the experimentation of the thermal camera from Raspberry pi we first enabled the I2C and VNC server and installed the pygame and scipy python script and run the script. The data received are shown in the next figure.

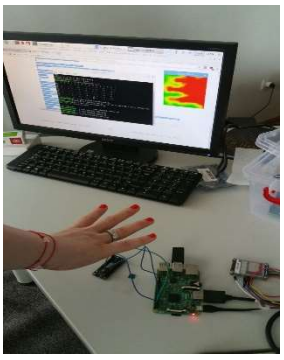

Fig. 10. Thermal image and temperature values

The major trends regarding the experimental consists of a:

- low latency, speed, and low bandwidth support;
- reduce the cost of communication, and provide better security of user data and information;
- reduce the physical interaction between people and increase health safety during COVID-19.

V. Conclusions

The main goal of this paper is to implement a multipoint video conferencing system through the Mozilla Firefox browser, where each user is connected to all other users, and the same video stream must be delivered to all connections. We also focus on the description of the video conference processing system. We have realized a video conference system using Scaledrone, which is a push messaging service, where you create a channel; you will find a unique ID that will utilize in the app. This is programmed in HTML, and the core of the system is a JavaScript API. To do this, we applied protocol ICE (Interactive Connectivity Establishment) together with the Session Traversal Utilities for NAT (STUN). An endpoint is aware only of its private address, and a parameter from another LAN (Local Area Network) will be unable to use this address for a connection. So, the STUN server is used by each endpoint to ask the public address that stands in front of the NAT (Network Address Translator). Now, the connections between public addresses are more comfortable to access.

WebRTC technology will be available through user's browsers to minimize installation and use of plugins in supporting communication. It will also improve the security of multimedia content and help developers to create better real-time video communication solutions.

The technology for video conference regarding the COVID-19 crisis will increase the use of the online meeting applications, together with an improvement in the domain of thermal imaging, which provides a healthy and secure life. The technology for video conference regarding the COVID-19 crisis will increase the use of the online meeting applications, together with an improvement in the domain of thermal imaging, which provides a healthy and secure life.

Acknowledgment

This paper has been partially supported by UEFISCDI Romania and MCI through project VIRTUOSE (Virtualized Video Services) and funded in part by European Union's Horizon 2020 research and innovation program under grant agreement No. 777996 (SealedGRID project) and No. 787002 (SAFECARE project).

References

[1] Nayyef, Zinah & Amer, Sarah & Hussain, (2019). Peer to Peer Multimedia Real-Time Communication System based on WebRTC Technology. International Journal for the History of Engineering & Technology. 2.9. 125-130.

[2] Suciu G., Anwar M., Mihalcioiu R.,Virtualized Video and Cloud Computing for Efficient e-Learning, 13th International Scientific Conference eLearning and Software for Education, April 27-28, 2017.

[3] Suciu G., Anwar M., Virtualized Video conferencing for eLearning, 14th International Scientific Conference eLearning and Software for Education Bucharest, April 19-20, 2018.

[4] Vasilescu C., Beceanu C., Collaborative object recognition for parking management, 15th International Scientific Conference eLearning and Software for Education Bucharest, April 11-12, 2019.

[5] Elleuch, Wajdi. (2013). Models for multimedia conference between browsers based on WebRTC. 279-284. 10.1109/WiMOB.2013.6673373.

[6] Rodríguez P, Cerviño J, Trajkovska I, Salvachúa J (2013) Advanced Videoconferencing Services Based on WebRTC. In: Proceeding of IADIS multi conference on computer science and information systems.

[7] XEP-0166: Jingle, XMPP Standards Foundation, https://xmpp.org/extensions/xep-0166.html.

[8] IC - Interactive Connectivity Establishment, IETF Working Group, https://tools.ietf.org/html/rfc5245.

[9] WebRTC Audio Codec and Processing Requirements, IETF Working Group, https://tools.ietf.org/html/rfc7874.

[10] "Why Does Your WebRTC Product Need a TURN Server?", https://www.callstats.io/blog/2017/10/26/turn-webrtc-products.

[11] Julius Flohr; Ekaterina Volodina; Erwin P. Rathgeb(2018) FSE-NG for managing real time media flows and SCTP data channel in WebRTC.

[12] Edim Azom Emmanuel; Bakwa Dunka Dirting (2017) A Peer-To-Peer Architecture For Real-Time Communication Using WebRTC.

[13] Vamis Xhagjika, Oscar Divorra Escoda, Leandro Navarro, Vladimir Vlassov(2017) Media Streams Allocation and Load Patterns for a WebRTC Cloud Architecture.

[14] Jansen, Bart & Goodwin, Timothy & Gupta, Varun & Kuipers, Fernando & Zussman, Gil. (2018). Performance Evaluation of WebRTC-based Video Conferencing. ACM SIGMETRICS Performance Evaluation Review. 45. 56-68. 10.1145/3199524.3199534.

[15] Sodhro Ali Hassan & Giancarlo Fortino Energy Management during Video Transmission in WBSNs", 14th IEEE International Conference on Networking, Sensing and Control (ICNSC), Calabria, Southern Italy, May 16-18, 2017.

[16] Sodhro Ali Hassan. Power Control Algorithms for Media Transmission in Remote Healthcare Systems, IEEE Access,Vol.6, July, 2018.