IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 11, Issue 11, December 2024 https://www.irjcs.com/archives

Video Conferencing and Live Streaming Platform Using Web RTC

Archana T. Singh
Department of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, India
Arnav Katiyar
Department of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, India
arnavkatiyar305@gmail.com
Anurag Yadav
Department of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, India
ay993218@gmail.com
Ankit Raj
Department of Information Technology,
Greater Noida Institute of Technology (Engg. Institute), Greater Noida, India
swartikr@gmail.com

Publication History
Manuscript Reference No: IRJCS/RS/Vol.11/Issue11/DCCS10085 | Research Article | Open Access | Double-Blind Peer Reviewed **Article ID:** IRJCS/RS/Vol.11/Issue11/DCCS10085
Received:28,November2024,Revised:04,December 2024 Accepted:10 December 2024 Published Online: 16, December 2024 http://www.irjcs.com/volumes/Vol11/iss-11/06.DCCS10085.pdf
Article Citation: Archana,Arnav,Anurag,Ankit (2024). Video Conferencing and Live Streaming Platform Using Web RTC. IRJCS: International Research Journal of Computer Science, Volume 11, Issue 10 of 2024 pages 666-670
doi:> https://doi.org/10.26562/irjcs.2024.v1111.06
BibTeX Archana@2024video

Abstract: People must communicate with one another for a variety of reasons, including personal and professional ones. Depending on the circumstances, an in-person meeting might not be feasible, which is more costly and logistically demanding. Additionally, the demand for long-distance and remote communication has increased as a result of the global epidemic. To address these issues, a web application for video conferencing is being implemented. Web RTC is used in the construction of the backend architecture. Web RTC is an open source, free framework that gives web browsers Real-Time Communications (RTC) features using straight forward Java script APIs. High-quality real-time communication applications for your webpage are the goal of Web RTC. The real-time web dynamically makes it possible to establish connections between browsers with ease. This makes it possible for our web application to offer a variety of functions, including audio calls, video chat, file sharing, screen sharing, and text chat. React JS is used to design the front end. Our signaling server is Firebase.

1. INTRODUCTION:

A technology called video conferencing enables people to host or attend meetings without physically being present. It is a tool for various platforms of communication. Features like screen sharing, recording, text messaging, and more might be offered by these platforms. Video conferencing is changing how we communicate with one another as a result of the Covid-19 pandemic and the rise in technology use. It is imperative that video conferencing be made simple to use and understand, particularly for those who are not tech-savvy. It must be easy for users to invite people, host or join meetings, and do so for many media device. Additionally, there should be no setup or installation required; the technology should be available straight from a browser. Particularly in order to lower human- caused risks, security and privacy are crucial. This application uses the Web RTC frame work, which may offer a web application for video conferencing that is safe, practical, and compatible with browsers on all platforms.

2. ENVIRONMENT AND RELATED STUDIES:

Zoom, Google Meet are just a few of the open-source and proprietary platforms that use Web RTC for real-time communication. Nevertheless, these platforms frequently encounter difficulties with resource optimization, security, and scalability. Numerous methods for improving Web RTC's performance are covered in the literature currently under publication; however there aren't many thorough implementations for live streaming and video conferencing combined.

ISSN: 2393-9842
https://www.irjcs.com/archives

3. DESIGN OF THE SYSTEM:

Structure

A Number of components make up Web RTC (Web Real-Time Communications), which facilitates real-time device communication:

P2P (peer-to-peer) Architecture

Participants can send data directly without an intermediary thanks to the architecture used by the majority of Web RTC Caps. Because it can continue to share data even if the server connection is lost, Web RTC is more widely used than previous communication methods.

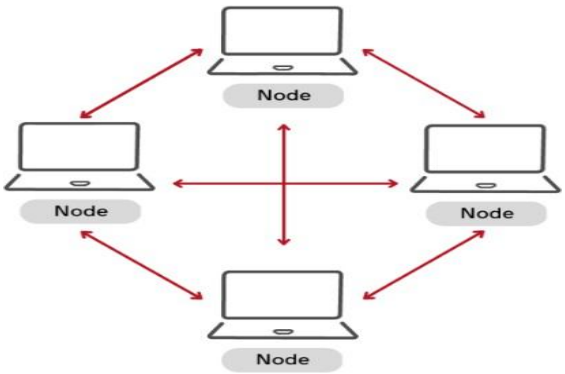

Fig.1.P2P architecture

The RTC Peer connection

This serves as the foundation of Web RTC and handles the majority of peer-to-peer communication duties. Once a peer connection is established, it enables the transmission of multimedia streams to a distant browser.

The m-DNS

This feature substitutes that are published and subscribed to throughout the LAN for IP addresses. It enables a connection between two peers without prior knowledge of each other's IP address.

The Media stream

Users can share audio and video information using this object. It comprises both audio and video that is received by the camera and microphone on a device.

RTC Data Channel

This data route is adaptable and can handle the delivery of both dependable and erratic data. It can be set up to meet different needs for data transport.

SDP

Devices that may be situated behind firewalls or network address translators (NATs) or have varying capabilities can communicate in real time thanks to this component.

4. LITERATURE SURVEY:

Real-time communication has been greatly enhanced by the quick development of web technologies. Due to its low-latency, peer-to-peer communication capabilities, Web RTC (Web Real-Time Communication) has become a crucial technology for developing video conferencing and live streaming systems. A summary of the current literature is provided here, emphasizing the key advancements, difficulties, and uses of Web RTC in live streaming and video conferencing.

Applications in Video Conferencing

Web RTC is used for real-time communication in video conferencing services such as Microsoft Teams, Zoom, and Google Meet. Key elements of Web RTC are highlighted in research by Singhetal.(2017):

Audio/video streams are established via RTC Peer Connection. Secure data transport is made easier using RTC Data Channel. Media Stream: Effectively manages streams of media. Important benefits are noted in the literature, such as flexible bit rate streaming, minimal latency, and support for encrypted communication. Scalability is still a problem, though. According to research like that conducted by Cordeiro et al.(2021), by selectively forwarding media streams rather than duplicating them, SFU (Selective Forwarding Unit) architecture can aid in the efficient management of numerous participants.

Applications in Live Streaming

Live broadcasting websites like Face book Live, YouTube Live, and Twitch also make considerable use of Web RTC. For interactive streaming scenarios like e-sports, online education, and virtual events, Web RTC is perfect since it has sub-second latency, unlike more conventional streaming protocols like RTMP. Web RTC's direct peer-to-peer communication can save server expenses significantly without sacrificing quality, according to research by Kumar and Sharma(2019). The peer-to-peer nature of Web RTC makes it difficult to achieve large-scale streaming. Research has looked into using hybrid architectures that combine Web RTC and CDNs (Content Delivery Networks) to manage large viewer ships with little latency.

Performance Metrics and Optimization: A number of factors affect how well Web RTC-based platforms work:

Latency: For the best user experience, real-time systems need to have negligible latency.

Network Resource Consumption: Effective bit rate control guarantees quality without using up too much bandwidth.

Service Quality, or QoS: Measures such as resolution, packet loss, and jitter are essential for preserving high-quality communication. Banerjee et al. (2020) conducted research that emphasizes how Web RTC's congestion control algorithms, including Google's Congestion Control (GCC), optimize resource utilization while providing consistent performance under a range of network situations.

Challenges and Limitations: Real-time communication has been transformed by Web RTC, yet it still faces a number of obstacles:

Scalability: As more people join, peer-to-peer networks close their effectiveness.

Security: Web RTC by default allows for encryption, but signaling server flaws can make systems vulnerable to attacks.

Despite being extensively supported, cross-browser compatibility might cause discrepancies due to variations in browser implementations.

Directions for the future: New developments in live streaming and Web RTC- based video conferencing centers on:

AI and ML Integration: Adding features like auto- framing, transcription, and real-time noise reduction to improve the user experience.

Edge computing: processing data closer to users to improve scalability and reduce latency.

Web Assembly (WASM): Enabling Web RTC applications to encode and decode more quickly and effectively.

5. WEBRTC TECHNICAL ASPECTS

Web RTC, or Web Real-Time Communication, is a robust open-source framework that eliminates the need for plugins by enabling real-time communication (RTC) features directly within web browsers and mobile applications. Its peer-to-peer (P2P) design facilitates the transmission of data, audio, and video, which makes it perfect for applications like interactive gaming, live streaming, and video conferencing. The technical details of Web RTC are covered in detail in this section, with particular attention paid to its architecture, constituent parts, protocols, and optimization methods.

Web RTC's Fundamental Elements: A Number of underlying technologies and three fundamental APIs that manage media capture, processing, and transmission form the foundation of Web RTC

The Media Stream API: Charged with recording and controlling audio and video streams from input devices like cameras and microphones. Gives developers the ability to adjust media parameters including bit rate, frame rate, and Resolution gives programmers control over media parameters including bit rate, frame rate, and resolution.

Connection API for RTC Peer: Facilitates Direct Communication between peers by controlling the flow of data and media between endpoints. Manages operations such as bandwidth negotiation, compression, and encryption. manages responsibilities such as bandwidth negotiation, encryption, and compression.

The RTC Data Channel API: Offers a dependable, low-latency route for peers to exchange arbitrary data. Helpful for file transfers, text message sending, game states, and more.

Web RTC Architecture: For connection establishment and maintenance, Web RTC's architecture uses a combination of signaling techniques and browser capabilities

Signalling: Web RTC depends on external systems (SDP, ICE candidates, etc.) to exchange connection information rather than defining Signaling protocols itself. Common protocols include HTTP and Web Socket. Steps for signaling discovery by peers. SDP is used to negotiate media capabilities. Sharing of NAT traversal information.

Interactions among Peers: In an effort to minimize latency and server load, Web RTC aims to link peers directly. Utilizing the Interactive Connectivity Establishment (ICE) architecture, NAT bypassing is accomplished by combining A peer's public IP address and port are determined via STUN (Session Traversal Utilities for NAT). When direct connections are unsuccessful, TURN (Traversal Using Relays across NAT) relays media via a middle server.

Managing Media: For encoding and decoding, Web RTC makes use of codecs such as VP8/VP9 (video), H.264 (video), and Opus (audio). For security, SRTP (Secure Real-Time Protocol) is used to encrypt media streams.

Essential Web RTC Protocols: Web RTC uses several protocols to provide safe and effective real-time communication

Protocol for describing sessions (SDP): Media communication parameters such as codec, bit rate, and resolution are described using a standard format.

Transport Protocol in Real Time (RTP): Allows peers to share video and audio streams. synchronizes by providing sequencing and time stamping.

SRTP, or Secure Real-Time Transport Protocol: Enables safe transmission by adding authentication and encryption to RTP packets.

Transport Layer Security for Data grams (DTLS): Enables the Web RTC handshake to securely transmit signaling data.

Layer Security of Transport (TLS): Encrypts information sent via signaling channels.

RTCP, or Real-Time Control Protocol: Gives feedback for congestion control and keeps an eye on the quality of the transmission.

Techniques for Web RTC Optimization: Web RTC uses a number of techniques to maximize efficiency in a range of network scenarios:

Management of Congestion: Adaptive Bit rate: This feature automatically modifies the quality of audio and video according to network bandwidth. Google's Congestion Control (GCC): Prevents network saturation by dynamically controlling transmitting rates.

Traversing a Network

The ICE architecture determines the most effective route to guarantee peer-to-peer connections are successful.

Expandability

To save bandwidth, a server uses a selective forwarding unit, or SFU, to forward certain streams, such active speaker. Streams are combined into a single feed by the MCU (Multipoint Control Unit), although this adds to the server load.

Error-Resilience

In order to preserve quality, packet loss concealment makes up for lost data packets. By adding redundant data, Forward Error Correction (FEC) makes it possible to recover lost packets.

Security in Web RTC

Web RTC (Web Real-Time Communication) is built on the foundation of security, guaranteeing that peer-to-peer real-time communication is secure and private. Its security paradigm tackles the inherent dangers of sending data, video, and audio via the internet. Here is a thorough breakdown of the main security measures that Web RTC uses:

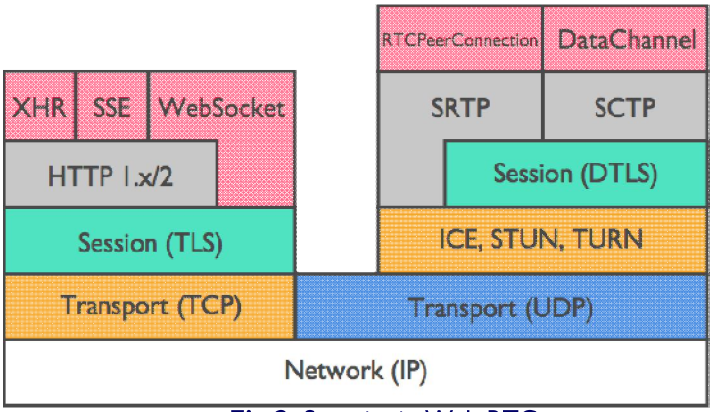

Fig.2. Security in Web RTC

Completely Secure Encryption: By using secure protocols, Web RTC makes sure that all data, audio, and video streams sent between peers are automatically encrypted. SRTP, or Secure Real-Time Transport Protocol used for video and audio stream authentication and encryption. Makes ensuring that information in media cannot be altered or intercepted while it is being transmitted. Transport Layer Security for Data grams (DTLS) establishes and maintains peer connections by protecting the signaling data. Stops control message manipulation and eaves dropping during Web RTC handshakes.

Identity Confirmation: The following features of Web RTC ensure that connections are made only with reliable entities by confirming the identification of communicating peers:

IdPs, or identity providers: To authenticate peers, Web RTC facilitates cooperation with external identity services. Guarantees the authenticity and authorization of a connection on both sides.

Certificate Trade: during the DTLS hand shake, a self- signed certificate is generated by each Web RTC peer. Prior to data sharing, these certificates confirm the legitimacy of peers.

Managing Permissions: In order to protect user privacy, Web RTC requires rigorous user consent before granting access to critical hardware resources:

Express Permissions- Browsers need the user's express consent to access screens, cameras, and microphones. Unauthorized access is prevented by granting permissions just for a single session.

Secure Context Needed- In order to prevent exploitation over unsecure connections, Web RTC functions are only available over HTTPS (secure web pages).

Security with NAT Traversal: Web RTC makes use of the ICE (Interactive Connectivity Establishment) architecture to provide peer-to-peer communication over various networks. This framework includes:

Servers that turn and stun: NAT traversal is made easier by TURN (Traversal Using Relays around NAT) and STUN (Session Traversal Utilities for NAT) servers. These servers just help create connections; they don't access or keep user data.

Limited Exposure to Information: By exchanging just necessary network data (such IP addresses), exposure to possible attackers is reduced.

6. CONCLUSION:

Through the ability to facilitate smooth peer-to-peer interactions directly within web browsers and applications without the need for plug-in, Web RTC (Web Real-Time Communication) has completely transformed real-time communication. Because of its low latency, strong functionality, and open-source nature, it has become a key technology for online gaming, live streaming, video conferencing, and other interactive platforms. Modern technologies like DTLS for data encryption, SRTP for secure media transmission, and the ICE framework for effective NAT traversal are all incorporated into Web RTC's architecture. Web RTC is appropriate for a variety of use cases because of these characteristics, which guarantee excellent performance, dependability, and security. The quality of service is improved even in difficult situations by its flexibility in responding to changing network circumstances, support for adaptive bit rate streaming, and congestion control.

ISSN: 2393-9842
https://www.irjcs.com/archives

REFERENCES:

1. Singh, H., Kakkar, D., & Gupta, M. (2017). "Web RTC: A Framework for Real-Time Peer-to-Peer Communication."International Journal of Computer Applications, 162(1), 1-6. Explores Web RTC's technical aspects and potential applications.
2. Cordeiro, W., Correa, T., & Cardoso, K. (2021). "Performance Analysis of Web RTC for Real-Time Communication."Journal of Network and Computer Applications, 173, 102879. Discusses scalability and performance issues in WebRTC implementations.
3. Banerjee, R., Prasad, K., & Jha, S. (2020). "Adaptive Bitrate Streaming and Congestion Control in Web RTC."IEEE Communications Surveys & Tutorials, 22(4), 2803-2825. A technical study on Web RTC's adaptive mechanisms. W3C (World Wide Web Consortium). WebRTC1.0: Real-Time Communication Between Browsers. Official specification document. URL: https://www.w3.org/TR/webrtc/
4. IETF (Internet Engineering Task Force). Interactive Connectivity Establishment (ICE) Framework. Provides technical details on the ICE mechanism used in Web RTC. URL: https://datatracker.ietf.org/
5. Mozilla Developer Network (MDN). Web RTC Documentation. A detailed guide on Web RTC APIs with examples. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API
6. Google Developers. Web RTC Overview and Tutorials. Hands-on tutorials and insights into Google's Web RTC implementation. URL: https://webrtc.org/
7. Git Hub Repository: Web RTC Samples and Demos. A collection of code samples and projects for learning and experimentation.URL: https://github.com/webrtc
8. Kumar, P., & Sharma, A. (2019). "Web RTC in Live Streaming Platforms: Challenges and Opportunities." ACM Transactions on Multimedia Computing, Communications, and Applications, 15(3s), 1-12.
9. Examines the role of Web RTC in live streaming and interactive platforms. Andriani, C., & Wang, L. (2020). "Scaling Real-Time Video Applications with Web RTC " IEEE Multimedia, 27(2), 25-34.
10. Discusses scalability solutions for Web RTC applications. Google. Web RTC for Business: Real-Time Communication in the Enterprise.
11. A white paper detailing the enterprise use cases and benefits of Web RTC. URL: https://webrtc.org/business/
12. Twilio Building Scalable Web RTC Applications Insights from Twilio's expertise in real-time communication platforms. URL: https://www.twilio.com/docs/webrtc

