International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 04, April 2026)

WebRTC: A Comprehensive Review of Real-Time Communication Technologies, Challenges and Future Direction

Abhinanth NA[1], Sudha D[2]
[1]Computer Applications, Semester 8, Student, SSTM(SCMS), Ernakulam, India
[2]Computer Applications, Asst. Professor, SSTM(SCMS), Ernakulam, India

***Abstract*— Web Real-Time Communication (WebRTC) has fundamentally changed how browsers handle real-time communication, making it possible for users to exchange audio, video, and data directly without needing any plugins. This paper brings together findings from recent academic work to give a broad picture of where WebRTC stands today what it does well, where it still struggles, and where it seems to be heading. The studies we reviewed point to several clear strengths: its open-source foundation, the security and speed it offers, and how naturally it fits into modern web development.That said, real problems remain around getting different systems to work together, handling large-scale deployments, and keeping up with the demands of varied network environments. We also look at some of the more exciting directions researchers are exploring, from running WebRTC in hybrid apps and 5G networks to using adaptive bitrate control and mixed reality.**

***Keywords*— Adaptive Bitrate Streaming, 5G Networks, Peer to Peer Communication, Real-Time Communication, Video Conferencing, WebRTC.**

I. Introduction

WebRTC lets browsers and mobile apps communicate directly with each other no plugins, no extra software. You've probably used it without realizing it, whether in a video call, an online class, a telehealth appointment, or a live stream. What makes WebRTC particularly useful is how it keeps delays low by handling the messy details of real-world networking through protocols like ICE, STUN, and TURN. It also slots neatly into the web technologies most developers already use, including HTML5, JavaScript, and Node.js.

II. Literature Review

A. A Systematic Review on WebRTC for Potential Applications and Challenges Beyond Audio Video Streaming

After working through 83 separate studies, Mahmoud and Abrozariba found that WebRTC has spread into far more areas than most people would assume well beyond standard video calls, into fields like telemedicine, augmented and virtual reality, and even gaming.

Researchers consistently point to its open-source codebase and the ease with which it connects to web APIs such as RTCDataChannel as two of its real strengths. That said, the review doesn't paper over the difficulties: the underlying architecture grows complicated quickly, legacy systems tend to cause friction, UDP introduces security concerns that need active management, and performance can deteriorate sharply once bandwidth runs short.

B. Design and Implementation of Protocol Conversion Server Based on GB/T 28181 and WebRTC

Bridging China's GB/T 28181 surveillance protocol with WebRTC is not a trivial task, and Jiang and Liu tackle it head-on. They designed a SIP proxy conversion server that functions as a go-between for the two protocols, enabling audio and video to move across platforms that would otherwise have no way to communicate. Testing with NodeJS and ZL Media Kit showed it performs adequately on a local network. The limitations they acknowledge are fair: wrapping the system in a translation layer adds complexity, and how it would behave under heavier, real-world network conditions has not yet been established.

C. A Hybrid Approach for WebRTC Video Streaming on Resource-Constrained Devices

Using WebRTC on a Raspberry Pi or an entry-level smartphone is a fundamentally different experience from using it in Chrome on a modern desktop, and this study puts concrete numbers behind that gap. When Diallo et al. compared standard browser-based WebRTC against hybrid implementations built with Electron and React Native, the hybrid models came out ahead on CPU consumption, memory footprint, and bandwidth usage consistently, not just in edge cases. For developers targeting IoT hardware or embedded systems, those gains matter considerably. The trade-off, though, is genuine: hybrid apps require more development effort, must be built separately for each target platform, and carry a steeper overhead when it comes to installation and ongoing maintenance.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 04, April 2026)

D. WebRTC over 5G: A Study of Remote Collaboration QoS in Mobile Environment

Nakazato et al. took WebRTC out into the streets of Tokyo to see how it actually performs on 5G — not in a lab, but in the kind of real urban environment where signals drop and handoffs happen constantly. Their findings show that mmWave delivers impressive quality with very low latency and jitter, while Sub6 and repurposed 5G bands are noticeably less stable when devices switch between towers. Using multiple mobile operators and spreading across frequency bands helps, but mmWave's short range and the messiness of real-world handovers still stand between this technology and widespread reliable deployment.

E. Performance Evaluation of WebRTC-Based Video Conferencing: A Comprehensive Analysis

Saini and Sharma built a WebRTC conferencing system on the MEAN stack with Socket.io and then put it under pressure with increasing numbers of users. At low to moderate loads, it holds up well latency stays low, packets get through, and DTLS with SRTP keep things secure. But as more users join, CPU usage climbs steeply and things start to slow down. Their conclusion is practical: pure peer-to-peer doesn't scale gracefully, and combining it with a client-server setup gives you much better control over resources when calls get bigger.

F. Mamba: Bringing Multi-Dimensional ABR to WebRTC

Mamba is an attempt to make WebRTC smarter about video quality by using multi-agent reinforcement learning. Rather than locking in fixed encoding settings, it watches what's happening on the network in real time and adjusts bitrate, resolution, and frame rate accordingly. The results are genuinely impressive up to a 16.9% improvement in VMAF video quality scores and noticeably lower delay compared to both default WebRTC and other learning-based approaches. Getting there isn't simple, though: training multiple reinforcement learning agents is computationally demanding, and the system needs careful tuning to get the best out of it.

G. Real-Time Rate Control of WebRTC Video Streams in 5G Networks: Improving QoE with Deep Reinforcement Learning

Smirnov and Tomforde trained a deep reinforcement learning agent to manage video bitrate in WebRTC over 5G, feeding it live data on jitter, round-trip time, and packet loss. The agent learns to make better decisions than Google's built-in congestion control, particularly when the network gets congested it adapts faster and squeezes out more throughput with less latency. The catch is the training process itself, which is computationally heavy.

The system also doesn't yet tap into device-level 5G information, which limits how well it can handle the specific quirks of 5G networks in practice.

H. A WebRTC-Based Framework for Real-Time Communication in Mixed Reality

Garcia et al. built RTC-MR to answer a fairly specific question: what would it take to get WebRTC working properly inside a mixed reality headset like the HoloLens 2? The framework they developed makes it meaningfully easier to create immersive communication apps, and they demonstrated it in a healthcare context through the Health 5G project. The design is modular and integrates without a lot of friction, which is a real plus. However, some of the underlying libraries it depends on are showing their age, mobile support is limited, and the signaling layer would need work before this could be used in a serious production setting.

I. Secure Dematerialization of Assessments in Digital Universities through Moodle, WebRTC and Safe Exam Browser (SEB)

Running secure online exams is a real problem in places where infrastructure can't be taken for granted, and Sylla et al. take it on by combining Moodle, WebRTC, and the Safe Exam Browser into a single integrated system. Before proposing anything new, they take stock of what has already been attempted strengthened quiz controls within Moodle, virtualization techniques meant to curb cheating and make a clear case for why those approaches, each on its own, still leave too much room for abuse. What they put forward is designed to close those gaps, with a particular focus on universities across Africa where budgets and infrastructure are tight, yet the pressure to deliver assessments that are both trustworthy and scalable is no less urgent than anywhere else.

J. Enhancing Latency Reduction and Reliability for Internet Services with QUIC and WebRTC

In his doctoral research, Li zooms out to ask a broader question: what would it actually take to use QUIC and WebRTC together to make internet services faster and more dependable? He starts from an honest accounting of where things currently fall short specifically, how DNS and anycast-based routing often stumble when different layers of the network stack are not in sync with one another. To address this, his work introduces edge computing and machine learning as tools for managing connections in a more responsive, dynamic way, using real-time video delivery as the primary yardstick for measuring whether the approach holds up in practice.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 04, April 2026)

K. Solving the Problem of Poor Internet Connectivity in Dhaka: Innovative Solutions Using Advanced WebRTC and Adaptive Streaming Technologies

Connectivity in Dhaka is genuinely poor by most measures, and Malinovskiy takes that as a starting point for asking what WebRTC can realistically deliver in that context. After mapping out the network problems and reviewing how adaptive streaming has been applied elsewhere, he develops a solution that combines dynamic transcoding with RTT-based interface switching and error correction. The result is a streaming system designed specifically to stay usable in congested conditions, backed by real network measurements rather than idealized assumptions.

L. Web-Based Networked Music Performances via WebRTC: A Low-Latency PCM Audio Solution

Playing music together in real time over the internet is one of those problems that sounds simple but turns out to be surprisingly hard. Sacchetto et al. dig into why existing tools like JackTrip and Soundjack have limitations and why standard WebRTC media streams introduce too muchlatency for musicians to actually play in sync. Their workaround sending raw PCM audio through WebRTC's DataChannel alongside the AudioWorklet API sidesteps some of those constraints and delivers noticeably better latency and audio quality for collaborative performance.

M. R-FEC: RL-based FEC Adjustment for Better QoE in WebRTC

WebRTC's default forward error correction is a blunt instrument it doesn't adapt to what's actually happening on the network, and earlier bitrate control work largely ignored the loss recovery side of things entirely. Lee et al. address both gaps with R-FEC, a reinforcement learning framework that adjusts video and FEC bitrates together, in real time. The improvement in perceived video quality under variable network conditions is significant, particularly in the kind of fluctuating environments that trip up static approaches.

N. Converge: QoE-driven Conferencing over WebRTC Multipath Video

Multipath protocols like MPTCP and MPQUIC seem like a natural fit for video conferencing, but Sathyanarayana et al. find that they consistently underperform in practice. The issue is that they weren't designed with video in mind. Converge is their answer: a WebRTC-based system that uses a scheduler which actually understands video characteristics, combined with a feedback loop that makes smarter decisions about which network paths to use and how much error correction to apply. The outcome is better throughput, lower latency, and a more consistent viewing experience compared to default WebRTC.

O. Enhancing WebRTC Covert Channels with Video Steganography for Internet Censorship Circumvention

Stegozoa is a censorship circumvention tool that hides data inside WebRTC video streams without anyone being able to tell it's there. Its predecessor Protozoa had a weakness: WebRTC gateways could detect it. Stegozoa fixes this by embedding data at the level of quantized DCT coefficients using least significant bit substitution and syndrome-trellis coding a technique that keeps visual quality intact while making the hidden channel much harder to spot. It also includes optimizations that prevent the traffic patterns from looking suspicious. The throughput is lower than you'd get with a direct connection, but for actually getting around censorship, it outperforms comparable tools in both security and adaptability.

TABLE I
Advantages and Disadvantages of WebRTC

Advantages	Disadvantages
Real-time audio, video, and data	No built-in signaling — requires custom implementation
Plugin-free browser support	NAT traversal issues — needs STUN/TURN servers
Open-source and royalty-free	Not easily scalable for large group calls
Cross-platform (Web, Mobile, Desktop)	Inconsistent behaviour across browsers
Encrypted and secure by default	Limited media control and codec customization

III. Types Of Security Issues In Webrtc

A. IP Address Leakage

Among the less obvious quirks of WebRTC is its tendency to expose a user's real IP address, even when they are connected through a VPN. The issue stems from how ICE candidate negotiation works: STUN servers disclose the public-facing IP, which can then be seen by the other party in the connection or by any malicious scripts embedded in the page [1], [15]. This is not a hypothetical edge case. A disclosed address can serve to track a user's location, identify them in politically sensitive contexts, or enable more targeted attacks. For applications where user privacy genuinely matters, this is a gap that cannot simply be ignored.

The most dependable fix is to route all media through a TURN server and avoid establishing direct peer connections altogether. When traffic flows through TURN, the remote party only ever sees the relay server's address never the user's own.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 04, April 2026)

Papers [4] and [15] each argue for TURN from different angles: [4] examines its value in network environments dominated by NAT, while [15] focuses on censorship circumvention situations where leaking an IP address could carry serious personal consequences. Paper [5] also endorses TURN as a sensible fallback that meaningfully strengthens both privacy and connection stability.

B. Denial of Service (DoS)

Because WebRTC relies on UDP and real-time signaling, it has limited natural defenses against flooding attacks. An adversarial client can overwhelm a signaling server or bury peers in rapid-fire connection requests, and there is no native throttling mechanism at the connection layer to stop this from happening [2], [5]. The system is vulnerable to multiple types of issue. For instance, too many invalid ICE candidates, a sudden spike in session join requests, or media streams that use excessive CPU can overload the system and cause it to fail What makes this particularly uncomfortable is that completing a media session is not even required — an attacker can simply push TURN or SFU servers into allocating resources until performance degrades for every other participant [5].

Among the practical defenses, rate-limiting the signaling server and requiring authentication before any call can be established stand out as straightforward starting points. Paper [2] puts this into practice within its protocol conversion server to guard against the kind of overload that could take the entire system offline. Paper [5] takes a complementary angle, recommending an SFU (Selective Forwarding Unit) architecture as a way to spread the load and reduce the system's exposure to brute-force resource exhaustion. Paper [13] adds something less obvious: the reinforcement learning component inside R-FEC is able to pick up on abnormal load patterns and respond by scaling back video quality, which narrows the window an attacker has to work within.

C. Eavesdropping

WebRTC uses DTLS-SRTP to protect its media streams, which is a reasonable foundation but protecting the media layer is not enough if the signaling channel is left exposed. When signaling travels over plain HTTP rather than HTTPS, anyone in a position to intercept traffic can capture SDP offers and ICE candidates and use them to mount a man-in-the-middle attack. Devices running outdated software or libraries further compound the risk, since known vulnerabilities in those environments may never have been addressed [1], [9].

The remedy is conceptually simple: enforce HTTPS or WSS for all signaling traffic, and configure DTLS-SRTP correctly on the media side. Paper [9] demonstrates this in a context where the stakes are high a remote examination system where both the signaling channel and the video feed between students and proctors must be kept secure. Paper [5] makes a more general point: end-to-end encryption covering both the signaling and media layers is not optional without it, a WebRTC session cannot be considered genuinely private.

D. Lack of Built-in Authentication

WebRTC has no built-in mechanism for verifying who is actually joining a session that responsibility falls entirely on the application layer. If a signaling server lacks proper identity checks, anyone who obtains a session link can potentially enter the call, impersonate a legitimate participant, or disrupt proceedings. In low-stakes settings this may be acceptable, but in telemedicine, remote education, or online assessments contexts where identity genuinely matters the absence of built-in authentication is a meaningful vulnerability [1], [2], [9].

The answer is to make authentication part of the signaling layer itself, using protocols like JWT or OAuth to confirm a user's identity before they are admitted to any session. Paper [9] does this in an online examination context by linking session access directly to existing Moodle accounts and reinforcing the browser environment with SEB. Paper [2] addresses a related challenge verifying that identities hold as sessions cross the boundary between SIP and WebRTC. Paper [1] takes the broadest view, arguing that authentication should not be treated as an add-on but as a core part of the architecture, something that must be in place before media negotiation can even begin.

IV. Conclusion

WebRTC has genuinely changed what's possible in browser-based communication, and the research reviewed here reflects how far it's spread beyond its original use case. It's now embedded in telemedicine systems, online classrooms, surveillance infrastructure, 5G mobile services, IoT devices, and mixed reality platforms. The openness of its codebase, how naturally it fits into the modern web stack, and its built-in support for secure, low-latency communication give it a flexibility that few comparable technologies can match.

That said, the picture isn't uniformly positive. The same body of research makes clear that real problems persist — the architecture can get complicated, older systems don't always cooperate, and performance can degrade badly in demanding network conditions. What's encouraging is that researchers are actively working on these fronts. Hybrid application models, deep reinforcement learning for congestion control, and adaptive bitrate streaming all show genuine promise as ways to push past the current limits.

As WebRTC keeps evolving, the most important work ahead is probably around robustness in unpredictable environments, clearer standardization for AR/VR and IoT integration, and making large-scale deployments actually manageable. The technology has already earned its place as a foundation for modern real-time communication — what comes next is making that foundation solid enough to hold up as the demands on it keep growing.

REFERENCES

[1] H. Mahmoud and R. Abozariba, "A Systematic Review on WebRTC for Potential Applications and Challenges Beyond Audio Video Streaming," in IEEE Access, vol. 10, 2022

[2] S. Jiang and J. Liu, Design and Implementation of Protocol Conversion Server Based on GB/T 28181 and WebRTC, in 2022 IEEE ICCECE, Guangzhou, China, 2022, pp. 324-328

[3] B. Diallo, A. Ouamri, and M. Keche, A Hybrid Approach for WebRTC Video Streaming on Resource-Constrained Devices, in 2021 ICICT, Coimbatore, India

[4] J. Nakazato et al, WebRTC over 5G: A Study of Remote Collaboration QoS in Mobile Environment, in 2021 IEEE NetSoft, Tokyo, Japan.

[5] S. S. Saini and L. S. Sharma, Performance Evaluation of WebRTC-Based Video Conferencing: A Comprehensive Analysis, in 2021.

[6] Y. Li, Z. Zhang, H. Chen, and Z. Ma, Mamba: Bringing Multi-Dimensional ABR to WebRTC, in Proc. ACM MM '22, Lisbon, Portugal, 2022.

[7] N. Smirnov and S. Tomforde, Real-time Rate Control of WebRTC Video Streams in 5G Networks, in 2022 IEEE GLOBECOM, Rio de Janeiro, Brazil.

[8] F. M. Garcia et al., A WebRTC-Based Framework for Real-Time Communication in Mixed Reality, Sensors, vol. 22, no. 19, pp. 7346, 2022.

[9] K. Sylla, B. Babou, and S. Ouya, Secure Dematerialization of Assessments in Digital Universities through Moodle, WebRTC and SEB, in Proc. CELDA, pp 259-266, 2022.

[10] X. Li, Enhancing Latency Reduction and Reliability for Internet Services with QUIC and WebRTC, Doctoral Dissertation, Aalto University, Finland, 2024

[11] P. Malinovskiy, Solving the Problem of Poor Internet Connectivity in Dhaka, arXiv preprint, arXiv:2506.17343, 2025.

[12] M. Sacchetto et al, Web-Based Networked Music Performances via WebRTC: A Low-Latency PCM Audio Solution, J. Audio Eng. Soc., vol. 70, no. 11, pp. 926-937, Nov 2022.

[13] I Lee et al, R-FEC: RL-based FEC Adjustment for Better QoE in WebRTC, in Proc. ACM MM '22, pp. 2948-2956, Oct 2022

[14] S. D. Sathyanarayana et al., "Converge: QoE-driven Multipath Video Conferencing over WebRTC," in Proc. ACM SIGCOMM 2023, pp. 637-653, Sep. 2023

[15] G. Figueira, D. Barradas, and N. Santos, Stegozoa: Enhancing WebRTC Covert Channels with Video Steganography, ASIA CCS '22, ACM, 2022