CGReplay++: Replaying Cloud Gaming Sessions to Evaluate UDP, QUIC, and SCReAM

Alireza Shirmarz
Universidade Federal de São Carlos
Sorocaba, SP, Brazil
ashirmarz@ufscar.br

Hugo Guilherme de Assis Paula
Universidade Federal de Goiás
Goiânia, GO, Brazil
hugo.guilherme@discente.ufg.br

Mateus de Nuzzi Bragatto
Universidade Federal de São Carlos
Sorocaba, SP, Brazil
mtsbrgtt@gmail.com

Kleber Vieira Cardoso
Universidade Federal de Goiás
Goiânia, GO, Brazil
kleber@ufg.br

Fábio Luciano Verdi
Universidade Federal de São Carlos
Sorocaba, SP, Brazil
verdi@ufscar.br

Abstract

Cloud gaming (CG) is highly sensitive to congestion because real-time game video is streamed to the gamer while user commands are sent back to the cloud server. However, fair transport evaluation is difficult since gameplay is non-deterministic and commercial CG platforms are closed-source. This demo presents *CGReplay++*, a congestion-aware extension of CGReplay for reproducible transport evaluation. CGReplay++ replays pre-captured CG sessions while preserving video, commands, timing, and bidirectional interactivity. It adds a configurable transport layer supporting UDP/WebRTC-style transport, QUIC, RTP over QUIC (RoQ), and SCReAM. The demo shows how the same gameplay session can be replayed under controlled network conditions to compare transport impact on gamer-side QoE/QoS metrics, including VMAF, SSIM, PSNR, response time, delay, jitter, loss, and throughput.

CCS Concepts

• **Networks → Network experimentation**; *Transport protocols.*

Keywords

Cloud Gaming Traffic Replay, Congestion Control Mechanism, Quality of Experience (QoE) Evaluation

ACM Reference Format:
Alireza Shirmarz, Hugo Guilherme de Assis Paula, Mateus de Nuzzi Bragatto, Kleber Vieira Cardoso, and Fábio Luciano Verdi. 2026. CGReplay++: Replaying Cloud Gaming Sessions to Evaluate UDP, QUIC, and SCReAM. In *Proceedings of the 34th ACM International Conference on Multimedia (MM '26), November 10–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3767308.3839015

1 Introduction and Motivation

Cloud gaming moves game execution and rendering from the gamer device to remote cloud or edge servers. The client mainly decodes the received game video and sends user commands back to the server [2, 7, 9]. This reduces the need for high-end local hardware, but it makes the experience strongly dependent on the network. Delay, jitter, packet loss, bandwidth limitation, and congestion directly affect both video quality and responsiveness [4, 8, 17].

Unlike conventional video streaming, CG is bidirectional and interactive. The downlink carries encoded game frames, while the uplink carries gamer commands. These two directions form a closed interaction loop: the gamer reacts to video frames, and the server updates the game state and generates future frames based on the received commands [4, 7]. Therefore, transport evaluation in CG must consider both video quality and response time.

A key challenge is *reproducibility*. Gameplay is naturally non-deterministic: different runs of the same game may produce different scenes, frame sizes, command timing, encoder behavior, and traffic patterns. Moreover, commercial CG platforms are closed-source, limiting control over the server, encoder, and transport stack. Prior studies show that CG platforms behave differently under changing network conditions and congestion [8, 15, 17]. This motivates a replay-based framework that keeps game content, commands, and timing fixed while changing only the network conditions and transport configuration.

This demo presents *CGReplay++*, a congestion-aware extension of CGReplay [12]. CGReplay++ replays pre-captured CG sessions as a controlled interaction between a gamer and a CG server. It preserves bidirectional timing, command events, video frames, and interactivity, while adding a configurable transport and congestion-control layer. CGReplay++ supports UDP/WebRTC transport[1], QUIC [3], RTP over QUIC (RoQ) [1], and SCReAM [5, 6]. This enables fair comparison of transport mechanisms under the same gameplay, user behavior, and network conditions.

2 CGReplay++ Design

2.1 Deterministic Cloud Gaming Replay

CGReplay++ builds on CGReplay, which captures real CG sessions and replays the interaction between the gamer and the server. In this proof-of-concept, we use traces collected from Microsoft Xbox Cloud Gaming. Each game profile contains the received game video, user commands, timing metadata, synchronization information, and PCAP traces. For the evaluated profiles, storage is approximately 200–242 MB/min, including video, command data, and PCAP traces,

MM '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2213-4/2026/11
https://doi.org/10.1145/3767308.3839015

[1]WebRTC includes signaling, session establishment, NAT traversal, security, media transport, and congestion-control components. In CGReplay++, we focus only on the media transport path, i.e., RTP/RTCP over UDP with WebRTC-style real-time transmission behavior, not the full WebRTC stack.

depending on the game. During replay, the command stream and video stream are synchronized at millisecond granularity to preserve the original action/reaction pattern.

This deterministic replay is the core idea of CGReplay++. Instead of playing the game again for each experiment, CGReplay++ reuses the same captured session. Therefore, different transport protocols can be compared using the same game content, command sequence, and timing behavior. This is important because CG traffic is content- and game-dependent: different games and scenes may generate different frame sizes, command rates, bitrate dynamics, and delay/loss sensitivity [8, 10, 16]. CGReplay++ intentionally reproduces the captured session rather than modeling server-side gameplay or encoder adaptations that would have occurred under a different network condition. This isolates transport effects but does not reproduce such closed-loop application adaptations.

2.2 Transport and Congestion-Control Layer

CGReplay++ extends the original UDP-based replay with a pluggable transport layer. The baseline follows a WebRTC-like media stack with H.264/H.265 video, RTP packetization, and UDP transport. CGReplay++ then adds QUIC, RoQ, and SCReAM as alternative transport/congestion-control mechanisms. QUIC is included as a modern UDP-based transport with congestion-control support [3]. RoQ enables RTP media over QUIC [1]. SCReAM targets self-clocked rate adaptation for real-time interactive media [5, 6].

The goal is not to claim one transport is always superior. Instead, CGReplay++ provides a fair environment to observe how each mechanism behaves under the same game, commands, video sequence, and network path. This is important because interactive media congestion control must keep delay low while maintaining useful throughput [4, 14].

2.3 Architecture

Figure 1 shows the CGReplay++ architecture. The system has three main components: the CG gamer, the CG server, and the configurable network path. The CG gamer receives and decodes the replayed H.264/H.265 video and sends commands according to the synchronization file. The CG server replays the captured video stream and receives the command stream with the reconstructed timing. Together, they preserve the original CG interactivity loop.

The network path is emulated using Mininet. CGReplay++ allows the user to configure bandwidth, delay, jitter, packet loss, and bottleneck conditions. The selected transport protocol and network configuration define the congestion behavior experienced by the replayed session.

CGReplay++ collects both QoS and QoE-related metrics at the gamer side. Network metrics include delay, jitter, packet loss, and throughput, while PCAP traces are also available. Video quality is measured using VMAF, SSIM, and PSNR, while interactivity is evaluated using response time between user commands and the corresponding video reaction. In this demo, we use three Xbox game profiles: *Forza*, *Fortnite*, and *Mortal Kombat*.

3 Novelty and Expected Impact

CGReplay++ extends CGReplay in three directions: deterministic replay of captured video, commands, and timing; pluggable evaluation

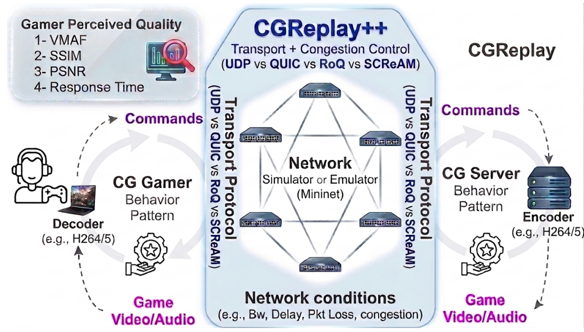

Figure 1: CGReplay++ architecture for reproducible cloud gaming transport evaluation over UDP/WebRTC-style transport, QUIC, RoQ, and SCReAM under controlled network conditions.

of UDP/WebRTC-style transport, QUIC, RoQ, and SCReAM under identical conditions; and joint reporting of network and gamer-side QoE metrics. It therefore provides researchers and developers with reproducible PCAP traces and QoE/QoS logs for assessing how transport choices affect cloud-gaming quality and responsiveness.

4 Demonstration

The demo replays three pre-captured Xbox game profiles—Forza, Fortnite, and Mortal Kombat—containing video, user commands, synchronization metadata, and PCAP traces. Attendees select the game, transport (UDP/WebRTC-style, QUIC, RoQ, or SCReAM), and network conditions, including bandwidth, delay, jitter, loss, and bottleneck settings. During replay, CGReplay++ reports VMAF, SSIM, PSNR, response time, delay, jitter, loss, and throughput. PCAP traces and QoE/QoS logs are stored after each run for offline comparison.

5 Conclusion and Future Work

CGReplay++ enables fair and reproducible cloud gaming transport evaluation by replaying the same captured game session while changing only the transport and network conditions. It compares UDP/WebRTC transport, QUIC, RoQ, and SCReAM game by game, enabling controlled analysis of RTP-based and QUIC-based delivery under congestion. The framework outputs PCAP traces, transport/game logs (bitrate, resolution, frame rate, packet size, and IPG), and gamer-side quality metrics such as VMAF, SSIM, PSNR, and response time.

Future work includes extending CGReplay++ to VR/XR cloud gaming, where stricter upstream command and downstream video requirements must be considered. We also plan to connect CGReplay++ with synthetic CG traffic generation [13], considering how transport and congestion-control behavior can affect bitrate, frame pacing, packetization, and payload characteristics. Finally, we will explore Agentic-AI-assisted generation to synthesize plausible video frames from game context and command traces [11], moving beyond replaying only one captured user behavior.

Acknowledgments

This work was supported by the Innovation Center, Ericsson S.A., and by the São Paulo Research Foundation (FAPESP), grant 2021/00199-8, CPE SMARTNESS.

References

[1] M. Engelbart et al. 2025. *RTP over QUIC (RoQ)*. Internet-Draft draft-ietf-avtcore-rtp-over-quic-13. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-over-quic/13/ Work in Progress.

[2] ITU-T. 2025. *Infrastructure Architecture Requirements for Cloud Gaming Service*. Recommendation ITU-T J.1312. International Telecommunication Union.

[3] Jana Iyengar and Martin Thomson. 2021. QUIC: A UDP-Based Multiplexed and Secure Transport. RFC 9000. (May 2021). https://doi.org/10.17487/RFC9000

[4] Randell Jesup and Zaheduzzaman Sarker. 2021. Congestion Control Requirements for Interactive Real-Time Media. RFC 8836. (Jan. 2021). https://doi.org/10.17487/RFC8836

[5] Ingemar Johansson and Zaheduzzaman Sarker. 2017. Self-Clocked Rate Adaptation for Multimedia. RFC 8298. (Dec. 2017). https://doi.org/10.17487/RFC8298

[6] Ingemar Johansson, Magnus Westerlund, and Mirja Kühlewind. 2026. *Self-Clocked Rate Adaptation for Multimedia*. Internet-Draft draft-ietf-ccwg-rfc8298bis-screamv2-00. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-ietf-ccwg-rfc8298bis-screamv2/00/ Work in Progress.

[7] Yang Li, Jiarong Qiu, Hao Wang, Zhenhua Li, Feng Qian, Junchen Yang, Haisheng Lin, Yunhao Liu, Bin Xiao, and Xin Qin. 2025. Dissecting and Streamlining the Interactive Loop of Mobile Cloud Gaming. In *Proceedings of the 22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association, 595–611.

[8] Minzhao Lyu, Yuxuan Wang, and Vijay Sivaraman. 2024. Do Cloud Games Adapt to Client Settings and Network Conditions?. In *Proceedings of the IFIP Networking Conference*. 482–488.

[9] Minzhao Lyu, Yifan Wang, and Vijay Sivaraman. 2026. Systematic Assessment of Cloud Game Adaptability for Network Conditions and User Experience. *Computer Networks* (2026).

[10] Saman S. Sabet, Steven Schmidt, Saman Zadtootaghaj, Carsten Griwodz, and Sebastian Möller. 2020. Delay Sensitivity Classification of Cloud Gaming Content. In *Proceedings of the ACM Workshop on Immersive Mixed and Virtual Environment Systems (MMVE)*. 25–30.

[11] Nedko Savov, Naser Kazemi, Mohammad Mahdi, Danda Pani Paudel, Xi Wang, and Luc Van Gool. 2025. Exploration-Driven Generative Interactive Environments. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*. 27597–27607.

[12] Alireza Shirmarz, A. G. de Castro, F. L. Verdi, and C. Rothenberg. 2025. CGReplay: Capture and Replay of Cloud Gaming Traffic for QoE/QoS Assessment. In *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*. https://netsoft2025.ieee-netsoft.org/

[13] Alireza Shirmarz, Goes de Castro Ariel, Victor Hugo Schneider Lopes, Fabio Verdi, Marcelo Luizelli, and Christian Rothenberg. 2025. CGSynth: Cloud Gaming Synthesizer. In *Proceedings of the ACM SIGCOMM 2025 Posters and Demos (ACM SIGCOMM Posters and Demos '25)*. Association for Computing Machinery, New York, NY, USA, 175–177. https://doi.org/10.1145/3744969.3748445

[14] Varun Singh and Zaheduzzaman Sarker. 2021. Evaluating Congestion Control for Interactive Real-Time Media. RFC 8868. (Jan. 2021). https://doi.org/10.17487/RFC8868

[15] Shuai Wang, Shuo Yang, Xiang Kong, Chuan Wu, Li Jiang, and Chengzhong Xu. 2024. Pudica: Toward Near-Zero Queuing Delay in Congestion Control for Cloud Gaming. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. 113–129.

[16] Yifan Wang, Minzhao Lyu, and Vijay Sivaraman. 2025. Games Are Not Equal: Classifying Cloud Gaming Contexts for Effective User Experience Measurement. In *Proc. ACM Internet Measurement Conference (IMC)*. Madison, WI, USA.

[17] Xiaokuan Xu and Mark Claypool. 2022. Measurement of Cloud-Based Game Streaming System Response to Competing TCP Cubic or TCP BBR Flows. In *Proceedings of the ACM Internet Measurement Conference (IMC)*. 305–316.