Receiving Kernel-Level Insights via eBPF: Can ABR Algorithms Adapt Smarter?

Mohsen Ghasemi[*], Daniele Lorenzi[†], Mahdi Dolati[*],
Farzad Tashtarian[†], Sergey Gorinsky[‡], Christian Timmerer[†]
[*]Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
[†]Christian Doppler Laboratory ATHENA, Alpen-Adria-Universität Klagenfurt, Austria
[‡]IMDEA Networks Institute, Spain

***Abstract*—The rapid rise of video streaming services such as Netflix and YouTube has made video delivery the largest driver of global Internet traffic including mobile networks such as 5G or the upcoming 6G network. To maintain playback quality, client devices employ Adaptive Bitrate (ABR) algorithms that adjust video quality based on metrics like available bandwidth and buffer occupancy. However, these algorithms often react slowly to sudden bandwidth fluctuations due to limited visibility into network conditions, leading to stall events that significantly degrade the user's Quality of Experience (QoE). In this work, we introduce CaBR, a Congestion-aware adaptive BitRate decision module designed to operate on top of existing ABR algorithms. CaBR enhances video streaming performance by leveraging real-time, in-kernel network telemetry collected via the extended Berkeley Packet Filter (eBPF). By utilizing congestion metrics such as queue lengths observed at network switches, CaBR refines the bitrate selection of the underlying ABR algorithms for upcoming segments, enabling faster adaptation to changing network conditions. Our evaluation shows that CaBR significantly reduces the playback stalls and improves QoE by up to 25% compared to state-of-the-art approaches in a congested environment.**

I. Introduction

Live streaming has become a major driver of video traffic, growing from under 1% to nearly 18% of all Internet traffic between 2015 and 2022 [1], [2]. Platforms like YouTube Live, Facebook Live, and Twitch highlight this demand.

HTTP Adaptive Streaming (HAS), including DASH [3] and HLS [4], remains the dominant delivery method for both VoD and live content due to its scalability and infrastructure compatibility.

HAS splits video into short segments, each encoded at multiple quality levels forming a bitrate ladder [5]. Clients download a manifest and adaptively select segments via an Adaptive Bitrate (ABR) algorithm (Figure 1). ABR algorithms are typically classified by the metrics used for selecting the next video segment [6]. Throughput-based approaches, like the `throughputRule` defined in the *dash.js* [7] player, estimate available bandwidth from recent downloads and select the highest bitrate that fits. While effective in stable networks, they often falter under fluctuating conditions. BOLA (Buffer Occupancy-based Lyapunov Algorithm) [8] combines buffer occupancy and bandwidth estimates, optimizing bitrate decisions via Lyapunov functions. It balances video quality and rebuffering risk, enabling stable performance in dynamic networks. LoL+ [9] targets low-latency streaming. It uses *k-means++* to initialize weights and self-organizing maps (SOM) to dynamically adapt QoE metrics. By adjusting the playback speed based on latency and buffer levels, it ensures smooth playback with low rebuffering in real-time scenarios.

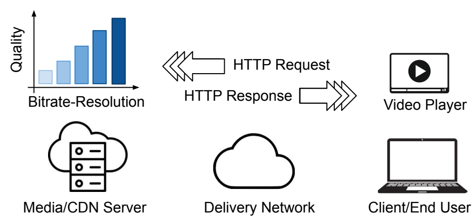
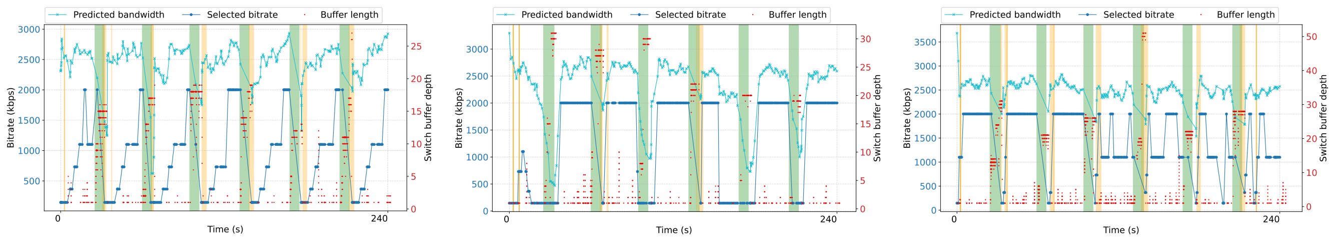


Fig. 1: Simplified video streaming pipeline.

Recent work has explored network-aware approaches to improve video streaming. AutoPlex [10] enables inter-session reuse of fine-tuned QUIC [11] and Bottleneck Bandwidth and Round-trip propagation time (BBR) [12] parameters for better QoE. Sultana et al. [13] leverage application-layer logic, Programming Protocol-independent Packet Processors (P4) [14], and network state for resource-aware adaptation. However, neither approach utilizes the extended Berkeley Packet Filter (eBPF) [15]. In contrast, several studies apply eBPF to improve network awareness. EyeQ [16] introduces a co-designed host-network system for responsive congestion control. Sundberg et al. [17] develop a kernel-level latency monitor for large-scale QoE inference. Herbots et al. [18] use eBPF to correlate transport- and application-layer metrics via TCP statistics. Qian et al. [19] apply eBPF to boost volumetric streaming resilience by duplicating packets across paths. Socker [20] enables socket-level integration for real-time application adaptation.

Building on this, we introduce CaBR, Congestion-aware adaptive BitRate, which enhances bitrate selection with transport-layer congestion metrics. CaBR tags packets with queue length at routers/switches via custom TCP headers, exposing this telemetry to the client through HTTP



(a) BOLA (b) LoL+ (c) Throughput-based ABR

Fig. 2: Comparison the performance of BOLA, LoL+, and throughput-based ABR algorithms.

requests. This enables more responsive ABR decisions, reduces rebuffering and quality fluctuations, and improves QoE under dynamic network conditions.

II. Motivating Example

To evaluate how ABR algorithms react to real-time congestion, we set up a controlled streaming session using the seqeunce Big Buck Bunny [21], segmented into 1 s chunks and encoded with AVC (*libx264*, *ultrafast*) following the bitrate ladder in [22]. The client maintains a 6 s playback buffer, which is typical for live streaming [23]. The video stream traverses three software switches (in the user space of the server); Switch #2 is selectively overloaded for 30 s intervals during a 4 min session to simulate transient congestion. Full testbed details are shown in Section IV. Figure 2 compares the performance of BOLA, LoL+, and throughput-based ABRs. Green regions show induced congestion; orange highlights playback stalls; red dots represent the queue lengths of Switch #2. All ABRs show delayed or inaccurate bitrate responses, failing to prevent stalls. Throughput-based ABR performs worst, often selecting high bitrates during congestion. BOLA and LoL+ behave more conservatively but still make poor decisions even in uncongested periods due to lack of awareness of network-level information. This reflects a core limitation: current ABRs rely solely on delayed application-layer metrics and do not have access to lower-layer congestion cues like queue length, especially due to browser sandboxing. These findings motivate our approach: using eBPF-based telemetry to expose real-time, kernel-level congestion signals (e.g., queue length) to the ABR alogrithm. This cross-layer insight enables more responsive bitrate adaptation, reducing stalls, and improving QoE under dynamic network conditions.

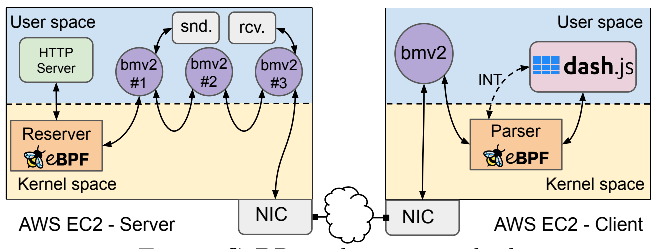


Fig. 3: CaBR evaluation testbed.

III. Proposed Approach

In this section, we describe our **C**ongestion-aware adaptive **B**it**R**ate algorithm (CaBR), which runs on top of underlying standalone ABR algorithms and refines their selected segment qualities based on collected real-time network telemetry. In Subsection III-A, we detail metric collection and ABR access, and Subsection III-B covers algorithmic use of telemetry data to improve quality.

A. Acquisition of Network-Level Metrics

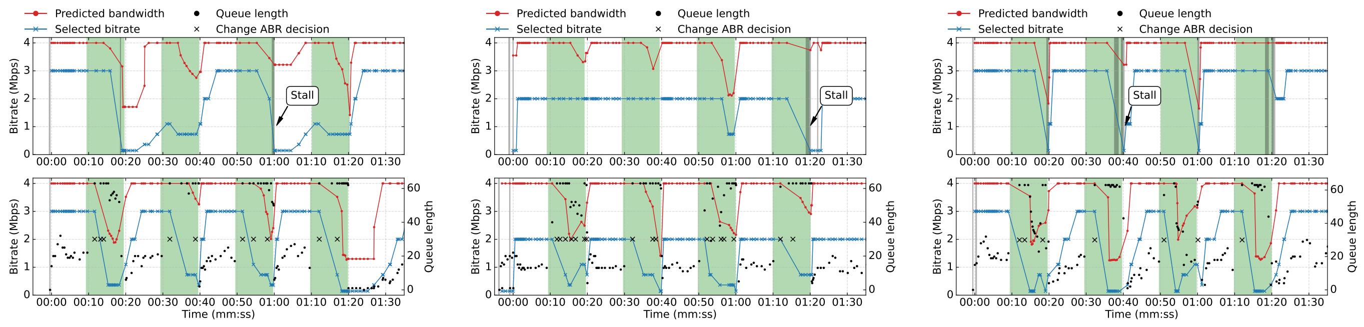
To collect network-level metrics, we adopt the In-band Network Telemetry (INT) [24] specification by introducing three functional entities into the video streaming framework: a *Source*, a *Sink*, and one or more *Transit Hops*. Our design implements the *Source* functionality at the server side using an eBPF program called *Reserver* that reserves a 4 B custom option within the TCP header of downstream packets.

Programmable network nodes located along the critical path of downstream packets function as *Transit Hops*. Each node parses the network telemetry-enabled packets and appends local metrics, specifically *queue length* and *packet-switching delay*, if it experiences a queue length above a predetermined threshold.

B. CaBR

To determine whether access to real-time congestion status influences QoE, we do not implement a standalone ABR algorithm from scratch, but rather adapt the bitrate selection of an underlying ABR algorithm according to low-level transport metrics, such as queue length.

a) eBPF-based Network Feedback: Each client continuously receives netowrk-layer informations like queue length via eBPF. The primary telemetry signal is the instantaneous queue length $Q \in [0, Q_{\max}]$ observed at the most congested router, where $Q_{\max}$ represents the router's maximum queue capacity (in packets of 1500 B). This signal acts as an early indicator of network congestion and can be used to refine bitrate decisions.



(a) BOLA (b) LoL+ (c) Throughput-based ABR

Fig. 4: Standard ABR algorithm (top) vs. ABR algorithm with CaBR (bottom).

b) Adaptation Logic: Let $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ be the bitrate ladder, defined as a set of discrete bitrates with $r_1 < r_2 < \cdots < r_n$. To express gradual bitrate adjustments, we introduce the notation:

$$(r_i \uparrow) := r_{\min(i+1,n)}, \quad (r_i \downarrow) := r_{\max(i-1,1)}.$$

which defines the immediate higher ($\uparrow$) and lower ($\downarrow$) bitrates with respect to r_i in the bitrate ladder $\mathcal{R}$, respecting the boundaries r_1 and r_n. CaBR adjusts the candidate bitrate $\tilde{R} \in \mathcal{R}$ selected by an underlying ABR algorithm for the next segment based on current queue length Q. The refinement relies on queue length thresholds $X < Y < Z$ in the range $[0, Q_{\max}]$ to categorize the congestion level into *safe*, *moderate*, *danger*, and *critical*, defining the final bitrate $\hat{R}$ as:

$$\hat{R} = \begin{cases} (\tilde{R}\uparrow), & 0 \le Q < X \quad \text{(Safe)} \\ \tilde{R}, & X \le Q < Y \quad \text{(Moderate)} \\ (\tilde{R}\downarrow), & Y \le Q < Z \quad \text{(Danger)} \\ ((\tilde{R}\downarrow)\downarrow), & Z \le Q \le Q_{\max} \quad \text{(Critical)} \end{cases}$$

Step-wise adjustments ensure gradual bitrate transitions, maintaining playback stability and avoiding bufferbloat.

IV. Performance Evaluation

We evaluate CaBR's effectiveness in enhancing ABR algorithms using eBPF-reported queue lengths. Experiments on Amazon EC2 [25] instances aim to assess (a) ABR behavior under congestion without CaBR, and (b) QoE improvements with CaBR integration.

Testbed Overview. Figure 3 shows our setup. (*i*) HTTPServer stores and streams the first 270 s of Big Buck Bunny in 1 s segments encoded via `libx264` using a standard bitrate ladder [22]. (*ii*) Switches bmv2, bmv2#1, bmv2#2, and bmv2#3 are Behavioral Model version 2 (BMv2) switches programmed with P4 with queue length $Q_{\max}$ = 64 packets, tail-drop policy, and 500 pps rate. (*iii*) The client runs the *dash.js* [7] player with a 6 s buffer and includes BOLA, LoL+, and Throughput-based ABR, with or without CaBR. (*iv*) A congestion generator exploits `iPerf` to induce periodic congestion via UDP cross-traffic between bmv2#1 and bmv2#3. Finally, (*v*) `sockops` programs monitor and log transport-level state via eBPF, while CaBR receives real-time telemetry through a cross-layer Go-based interface.

Scenarios. Each 100 s test runs one of the six ABR configurations. Starting at 10 s, four 10 s congestion bursts are introduced, each followed by a 10 s idle period. Each scenario is repeated five times.

Results. Figures 4a–4c show the performance of BOLA, LoL+, and Throughput-based ABRs with (bottom) and without (top) CaBR. Each figure presents predicted bandwidth (red), selected bitrate (blue), congestion periods (light green), playback stalls (dark green), queue lengths (black circles), and CaBR-triggered decisions (black crosses). In all cases, CaBR enables early congestion detection and proactive bitrate adjustments, completely preventing stalls. In contrast, baseline ABRs react too late, often mid- or post-congestion, resulting in up to six stalls. Figure 5 summarizes QoE using the ITU-T P.1203 model with Mode 0 [26], [27], normalized to [0, 1] based on stall metrics, average bitrate, and quality switches. CaBR consistently improves QoE across all ABRs by up to 25%. BOLA, which relies on buffer occupancy, responds more slowly to sudden congestion than LoL+ or throughput-based ABR. With CaBR, BOLA benefits from real-time congestion signals, avoids rebuffering, and achieves smoother playback.

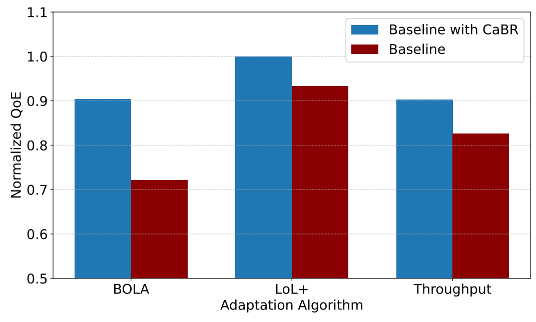


Fig. 5: Normalized QoE of baseline ABR algorithm vs. baseline ABR with CaBR.

V. Conclusion

This paper presents CaBR, a congestion-aware bitrate regulation mechanism that operates on top of existing ABR algorithms. CaBR leverages cross-layer, real-time network telemetry, collected via eBPF, to rapidly detect congestion and refine bitrate selection decisions. Experimental results across three representative ABR algorithms – BOLA, LoL+, and throughput-based ABRs – demonstrate that CaBR provides a smooth playback with no stalls and improves overall QoE by up to 25%, compared to the baselines operating alone. Ongoing work examines a broader range of video sequences, segment durations, network conditions, and ABR strategies to further validate and generalize the approach.

Acknowledgments

The research was supported in part by the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology, and Development, and the Christian Doppler Research Association through the Christian Doppler Laboratory ATHENA (https://athena.itec.aau.at/). Additionally, it was supported by project PID2022-140560OB-I00 (DRONAC), funded by MICIU/AEI/10.13039/501100011033, ERDF, and EU.

References

[1] SkyQuest Technology. (2022) Global online video platforms market drives over 80% of total internet traffic. [Online] Available: https://www.globenewswire.com/news-release/2022/08/02/2490661/0/en/Global-Online-Video-Platforms-Market-Drives-over-80-of-Total-\\Internet-Traffic-Skyquest-Technology.html. Accessed: April 22, 2025.

[2] Sandvine, "2023 global internet phenomena report," 2023, https://www.sandvine.com/hubfs/Sandvine_Redesign_2019/Downloads/2023/reports/Sandvine\%20GIPR\%202023.pdf. Accessed: April 22, 2025.

[3] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming over the Internet," *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, 2011.

[4] R. Pantos and W. May, "HTTP Live Streaming. RFC 8216," [Online] Available: https://www.rfc-editor.org/info/rfc8216, August 2017, accessed: 31 January 2025.

[5] T. Stockhammer, "Dynamic adaptive streaming over http –: standards and design principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11. Association for Computing Machinery, 2011. [Online]. Available: https://doi.org/10.1145/1943552.1943572

[6] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2019.

[7] DASH Industry Forum, "DASH Reference Player," [Online] Available: https://reference.dashif.org/dash.js/, 2024, accessed: 31 January 2025.

[8] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-Optimal Bitrate Adaptation for Online Videos," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1698–1711, 2020.

[9] A. Bentaleb, M. N. Akcay, M. Lim, A. C. Begen, and R. Zimmermann, "Catching the Moment With LoL$^+$ in Twitch-Like Low-Latency Live Streaming Platforms," *IEEE Transactions on Multimedia*, vol. 24, pp. 2300–2314, 2022.

[10] B. Wu, T. Li, C. Luo, C. Ouyang, X. Du, and F. Wang, "Autoplex: inter-session multiplexing congestion control for large-scale live video services," in *Proc. ACM SIGCOMM Workshop on Network-Application Integration*, 2022, p. 1–6.

[11] I. E. T. F. (IETF), "Quic: A udp-based multiplexed and secure transport," [Online] Available: https://datatracker.ietf.org/doc/html/rfc9000, 2021, accessed: April 22, 2025.

[12] I. C. C. R. Group, "Bbr congestion control," [Online] Available: https://www.ietf.org/archive/id/draft-cardwell-iccrg-bbr-congestion-control-01.html, 2021, accessed: April 22, 2025.

[13] N. Sultana, "Leveraging in-network application awareness," in *Proc. ACM SIGCOMM Workshop on Network-Application Integration*, 2021, p. 63–67.

[14] P4, "P4 open source programming language," 2025, accessed: April 22, 2025. [Online]. Available: [Online]Available:\url{https://p4.org/}

[15] eBPF Open Source Community, "ebpf - introduction, tutorials & community resources," [Online] Available: https://ebpf.io/, 2025, accessed: April 22, 2025.

[16] J.-T. Hinz, V. Addanki, C. Györgyi, T. Jepsen, and S. Schmid, "Tcp's third eye: Leveraging ebpf for telemetry-powered congestion control," in *Proceedings of the 1st Workshop on EBPF and Kernel Extensions*. ACM, 2023, p. 1–7.

[17] S. Sundberg, A. Brunstrom, S. Ferlin-Reiter, T. Høiland-Jørgensen, and J. Dangaard Brouer, "Efficient continuous latency monitoring with ebpf," in *Passive and Active Measurement*, 2023, pp. 191–208.

[18] J. Herbots, M. Wijnants, W. Lamotte, and P. Quax, "Cross-layer metrics sharing for quicker video streaming," in *Proc. International Conference on Emerging Networking EXperiments and Technologies*, 2020, p. 542–543.

[19] P. Qian, N. Wang, F. C. Heng, J. Zhang, C. Udora, and R. Tafazolli, "Enabling ebpf-based packet duplication for robust volumetric video streaming," in *2024 IEEE Symposium on Computers and Communications (ISCC)*, 2024, pp. 1–7.

[20] D. Guo, S. Wang, and Y. R. Yang, "Socker: Network-application co-programming with socket tracing," in *Proc. ACM SIGCOMM Workshop on Network-Application Integration*, 2021, p. 14–19.

[21] Blender Foundation, "Big Buck Bunny," [Online] Available: https://bit.ly/3YqoaZu, 2008, accessed: 31 January 2025.

[22] Apple, "HLS Authoring Specification for Apple Devices," [Online] Available: https://developer.apple.com/documentation/http_live_streaming/hls_authoring_specification_for_apple_devices, June 2020, accessed: 31 January 2025.

[23] F. Tashtarian, M. Dolati, D. Lorenzi, M. Mozhganfar, S. Gorinsky, A. Khonsari, C. Timmerer, H. Hellwagner *et al.*, "ALPHAS: Adaptive Bitrate Ladder Optimization for Multi-Live Video Streaming," in *IEEE International Conference on Computer Communications*, 2025.

[24] T. P. A. W. Group, "In-band network telemetry (int) dataplane specification," [Online] Available: https://github.com/p4lang/p4-applications/blob/master/docs/telemetry_report.pdf, accessed: April 22, 2025.

[25] Amazon, "Amazon Elastic Compute Cloud (Amazon EC2)," [Online] Available: https://aws.amazon.com/ec2/, accessed: 31 January 2025.

[26] W. Robitza, S. Göring, A. Raake, D. Lindegren, G. Heikkilä, J. Gustafsson, P. List, B. Feiten, U. Wüstenhagen, M.-N. Garcia, K. Yamagishi, and S. Broom, "HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 – Open Databases and Software," in *9th ACM Multimedia Systems Conference*, Amsterdam, 2018.

[27] A. Raake, M.-N. Garcia, W. Robitza, P. List, S. Göring, and B. Feiten, "A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1," in *Ninth International Conference on Quality of Multimedia Experience (QoMEX)*. Erfurt: IEEE, May 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7965631/