©2024 IEEE. This is the authors' accepted version of the article. The original publication is to appear in IEEE Xplore: 13th IEEE International Conference on Cloud Networking (CloudNet 2024).

In-Band Quality Notification from Users to ISPs

Leonardo Peroni
UC3M, Spain

Sergey Gorinsky
IMDEA Networks Institute, Spain

Farzad Tashtarian
Alpen-Adria Universität Klagenfurt, Austria

***Abstract*—While ISPs (Internet service providers) strive to improve QoE (quality of experience) for end users, end-to-end traffic encryption by OTT (over-the-top) providers undermines independent inference of QoE by an ISP. Due to the economic and technological complexity of the modern Internet, ISP-side QoE inference based on OTT assistance or out-of-band signaling sees low adoption. This paper presents IQN (in-band quality notification), a novel mechanism for signaling QoE impairments from an automated agent on the end-user device to the server-to-client ISP responsible for QoE-impairing congestion. Compatible with multi-ISP paths, asymmetric routing, and other Internet realities, IQN does not require OTT support and induces the OTT server to emit distinctive packet patterns that encode QoE information, enabling ISPs to infer QoE by monitoring these patterns in network traffic. We develop a prototype system, YouStall, which applies IQN signaling to ISP-side inference of YouTube stalls. Cloud-based experiments with YouStall on YouTube Live streams validate IQN's feasibility and effectiveness, demonstrating its potential for accurate user-assisted ISP-side QoE inference from encrypted traffic in real Internet environments.**

***Index Terms*—End user, QoE, ISP, QoE-impairment inference, OTT provider, end-to-end traffic encryption, in-band signaling.**

I. Introduction

ISPs (Internet service providers) have a vested interest in improving QoE (quality of experience) for end users. Even mid-path ISPs, which do not have a direct relationship with end users, aim to enhance QoE to maintain a positive business reputation, attract and retain customers and peers, and remain competitive in Internet connectivity markets. The means available to ISPs for mitigating QoE impairments include traffic shaping, flow prioritization, and rerouting [1], [2].

To mitigate QoE impairments, ISPs first have to infer them. Traditionally, ISPs infer QoE independently through techniques such as per-flow DPI (deep packet inspection), which examines the payloads of packets flowing through their routers. However, OTT (over-the-top) providers of streaming and other services increasingly adopt end-to-end traffic encryption. For example, YouTube utilizes the QUIC (quick UDP Internet connections) protocol [3] to encrypt its traffic. Because end-to-end encryption renders DPI-based QoE inference ineffective, researchers propose various alternative methods for extracting QoE insights from encrypted traffic [4]–[11]. Despite the extensive research, independent ISP-side solutions still face challenges in achieving high performance.

One way for an ISP to improve on independent QoE inference is by receiving assistance from the OTT provider. OTT providers run their clients on end-user devices or use web browsers to engage directly with users through interactive interfaces. OTT hypergiants like Netflix leverage vast cloud and edge infrastructures and operate their own private wide-area networks to stream content from globally distributed locations [12]. While OTT providers are well-positioned to support ISP-side QoE inference with existing technical capabilities [13], [14], they rarely do so in practice. Instead, OTT providers often cite fairness concerns to advocate for network neutrality regulations [15], which typically restrict ISPs from engaging in application differentiation, except during transient congestion, and generally limit them to supplying basic Internet connectivity.

Whereas OTT providers' reluctance to assist ISPs in QoE inference is understandable from the "tussle in cyberspace" perspective [16], end users have a clear incentive to help ISPs improve QoE for specific applications [17]. Existing solutions for user-assisted QoE inference include ALTO (application-layer traffic optimization) [18] and P4P (proactive network provider participation for P2P) [19]. These proposals support out-of-band signaling of QoE information from an end user to an ISP but do not gain widespread adoption.

To understand the low adoption of out-of-band QoE signaling, one has to consider the complex realities of the modern Internet. The tiered Internet structure contains thousands of ISPs providing global connectivity for billions of end users who consume OTT services. The complex structure features multi-ISP paths [14] and weaves together entities of different kinds via diverse but typically bilateral relationships.

Figure 1 illustrates a representative scenario where a congested tier-2 ISP acts as a transit customer [20] of a tier-1 ISP [21], buys partial transit [22] from another tier-1 ISP, sells transit to two tier-3 ISPs, and peers with a tier-2 ISP via a private interconnection [23]. It also purchases a remote-peering service [24] to reach an IXP (Internet exchange point) [25] and publicly peers at this IXP with another tier-2 ISP. ISP-level routing is asymmetric, and an OTT service operates its global private network, receiving client requests through one ISP and responding through another. When congestion occurs in a mid-path ISP and impairs QoE, the end user cannot determine which ISP to contact through out-of-band signaling to mitigate the congestion [26]. Additionally, the congested mid-path ISP is unaware of the affected users, and hence cannot contact them via out-of-band communication, because NAT (network address translation) [27] alters network flow identifiers.

While the economic and technological complexities of the modern Internet undermine adoption of seemingly well-grounded approaches, such as OTT-assisted inference and out-of-band signaling of QoE, this paper explores a user-assisted ISP-side QoE inference alternative that relies on *IQN (in-band*

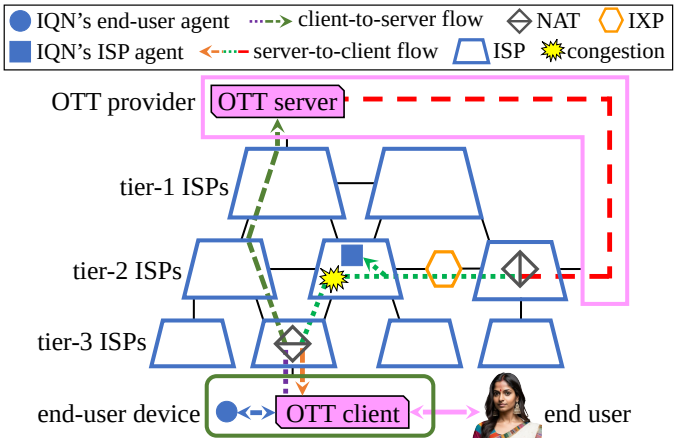


Fig. 1: IQN signaling of QoE from the end user of an OTT service to server-to-client ISPs in the economically and technologically complex Internet ecosystem.

quality notification), a novel mechanism for in-band signaling of QoE from an automated end-user agent to server-to-client ISPs, without requiring any OTT support. We derive the IQN design from principles of OTT independence, user simplicity, and Internet compatibility. Our chief insight is that OTT clients offer interactive features that allow an application on the end-user device to programmatically trigger a distinct pattern in server-to-client transmissions. Leveraging this insight, IQN's end-user agent induces the OTT server to transmit distinctive packet patterns encoding QoE, and a server-to-client ISP runs IQN's automated ISP agent in its routers to infer the encoded QoE information by observing these patterns in monitored network traffic. Akin to SOS, an IQN signal serves as an emergency notification to unknown ISPs capable of mitigating transient QoE-impairing congestion.

To assess the feasibility and performance of IQN signaling in a real Internet environment, we instantiate IQN in a system prototype. After our preliminary studies show IQN's effectiveness for three major OTT services and different QoE factors, we develop a prototype for inferring stalls of the QUIC-based YouTube service. This system, named *YouStall*, emits IQN signals by toggling the autoplay switch in YouTube's interface. The prototype incorporates an Amazon EC2 (elastic compute cloud) instance [28], which emulates an ISP router executing IQN's ISP agent to infer stalls and estimate their duration. IQN-assisted QoE inference at the router incurs less overhead compared to the traditional independent ISP-side QoE inference based on per-flow data-plane DPI. Additionally, YouStall leverages IQN's end-user agent for user-side stall detection by periodically capturing screenshots and identifying the spinning icon in YouTube's playback area.

Our experiments with YouStall on YouTube Live [29] streams of three genres corroborate the promise of IQN signaling. For example, while the average stall duration across the three genres exceeds 1.4 s, IQN-assisted ISP-side inference estimates the duration of significant stalls (those lasting at least 400 ms) with an average MAE (mean absolute error) and RMSE (root mean square error) of 231 and 288 ms, respectively. The experiments also evaluate YouStall's parameter sensitivity and overhead.

Overall, this paper makes the following main contributions:

1) We propose IQN, a novel mechanism for automated in-band signaling of QoE from the end user of an OTT service to server-to-client ISPs responsible for QoE-impairing congestion. IQN supports accurate ISP-side QoE inference despite end-to-end traffic encryption, asymmetric routing, and other Internet realities.
2) The paper presents YouStall, a system prototype that instantiates IQN by toggling YouTube's autoplay switch, enabling ISP-side inference of YouTube stalls.
3) Through cloud-based experiments on YouTube Live streams, we show IQN's feasibility and effectiveness.

II. Motivation and Principles

Our paper proposes a novel mechanism for ISP-side QoE inference. Although OTT providers, which directly interact with end users, are in the best position to assist ISPs in this task, real-world evidence shows little interest from OTT services in cooperating with ISPs on QoE measurement and improvement. On the contrary, OTT providers often complicate independent QoE inference by an ISP through actions like end-to-end traffic encryption. Given this reality, we argue that an effective ISP-side QoE inference mechanism should not expect support from the OTT service, leading to our first design principle:

Principle 1 (OTT independence): The mechanism should operate without any support from the OTT provider.

Since end users are the primary beneficiaries of QoE improvement, they have an incentive to assist ISP-side QoE inference [17]. However, this assistance should align with the low-effort role typical of casual users and avoid placing a significant burden on them:

Principle 2 (User simplicity): The effort required by the mechanism from the end user should be minimal.

While we allow for limited support from the end user, the complex structure of the Internet poses practical challenges for leveraging this support in ISP-side QoE inference. As Figure 1 illustrates, NAT might cause a network flow to change its identity as it traverses the Internet. Due to asymmetric inter-domain routing in the tiered Internet structure with multi-ISP paths [14], an OTT server might receive client requests through one ISP and respond through another. Consequently, the ISP responsible for QoE-impairing congestion, and capable of resolving the issue, might be off the client-to-server path. This requires end-user support to reach ISPs along the server-to-client path. Hence, the QoE inference mechanism should account for such practical Internet constraints:

Principle 3 (Internet compatibility): The mechanism should function effectively within the realities of the current Internet, such as multi-ISP paths, asymmetric routing, and NAT.

III. In-band quality notification

A. General IQN mechanism

The design principles outlined in Section II guide our novel approach to QoE inference by an ISP. While mainstream efforts pursue independent ISP-side QoE inference, they struggle to accurately infer QoE from encrypted traffic [4], [5]. As per Principle 1, OTT providers are unlikely to remove encryption or assist ISPs in inferring QoE through other means. Therefore, we explore *ISP-side QoE inference assisted by the end user*.

The key issue addressed in this paper is QoE signaling from end users to ISPs. Although ALTO [18], P4P [19], and other cross-layer designs explicitly support such signaling, we attribute their low adoption to a mismatch between their out-of-band nature and the Internet's structural complexity. With multiple tiers of ISPs, multi-ISP paths, and congestion occurring before the last-hop ISP, the end user is typically unaware of which ISP is responsible for QoE-impairing congestion [26]. Similarly, an ISP is often unaware of the specific end users due to NAT and other modern Internet realities. Since the end user generally does not know which ISPs to notify about QoE impairments via an out-of-band mechanism, our research focuses on *in-band signaling*.

According to Principle 3, the signaling mechanism should be compatible with asymmetric routing and notify ISPs along the server-to-client path of the OTT service. One possibility is for the OTT server to reflect in-band signals received from the end user, similar to how network cookies facilitate network neutrality [17]. However, this support from the OTT provider would violate Principle 1. Hence, we seek a mechanism for *in-band signaling to server-to-client ISPs without relying on OTT support*.

End users typically exert low effort when consuming services, often limited to installing and running applications like an OTT client. For example, ECN (explicit congestion notification) [30], which transmits in-band congestion signals from networks to end devices for transport-layer reactions, operates transparently to end users and their applications. In conformity with Principle 2, we aim to maintain this minimal level of effort, requiring *only the installation of an agent on the end-user device, which automatically signals QoE*.

The principle-guided approach leads to the design of *IQN*, a new mechanism for in-band signaling of QoE information from an automated end-user agent to server-to-client ISPs, without relying on OTT support. IQN's key insight is that each major OTT service provides a rich interactive interface, allowing the end user to issue commands through the OTT client and receive responses from the OTT server. IQN's end-user agent leverages this interface to programmatically send a series of commands that induce the OTT server to transmit a distinctive packet pattern, aligned with the OTT service's internal logic, to the client. This packet pattern encodes QoE information, which the server-to-client ISPs infer by detecting the pattern in the network flow.

While this paper primarily focuses on QoE signaling from the end user to server-to-client ISPs, which is our most innovative contribution, we also examine other, less novel but important, aspects of user-assisted ISP-side QoE improvement. For example, the end-user agent automatically detects QoE by leveraging again the OTT client's interface.

For ISP-side mitigation of inferred QoE impairments, we consider well-established techniques such as traffic prioritization [1] and rerouting [2]. For example, when an ISP infers a QoE impairment in a network flow on a congested link, it assigns higher priority to that flow. If multiple server-to-client ISPs infer the same impairment, they respond independently based on their own understanding of potential local causes, without coordination. IQN acts like a distress signal to any ISP capable of resolving the transient QoE-impairing congestion, an emergency rather than a persistent issue.

B. IQN instance in the YouStall system

To evaluate the feasibility and accuracy of the IQN mechanism in real Internet environments, we instantiate it in a system prototype. Our preliminary analysis of Amazon Prime Video, Netflix, and YouTube reveals that their client interfaces offer interactive features, such as an autoplay switch and audio language selection, that support effective IQN signaling. In line with the scenarios targeted by IQN, we develop the prototype for YouTube, which encrypts its end-to-end traffic in QUIC packets. The system notifies ISPs about playback stalls, a prominent QoE influence factor [31]. We refer to the prototype as YouStall.

User-side QoE signaling. To issue IQN signals via YouTube's interface, YouStall utilizes the autoplay switch, which determines whether the next video plays automatically after the current one finishes. We select this feature to avoid disrupting the user experience. The end-user agent emits an IQN signal by toggling the autoplay switch programmatically an even number of times, with intervals shorter than 5 ms between toggles. This sequence of n toggles occurs too quickly for human perception and concludes with both the cursor and autoplay switch in their original positions. By default, we set n to 4 as the smallest even number that is unlikely to occur naturally in user behavior while still triggering a distinctive pattern in server-to-client transmissions.

The end-user agent issues an n-toggle IQN signal whenever its QoE check, conducted at intervals of p (set to 200 ms by default), detects a playback stall. For each toggle in the IQN signal, YouTube's client transmits an HTTP/3 (hypertext transfer protocol version 3) POST request to the YouTube server, which updates its autoplay settings and responds with an HTTP/3 "200 OK" status over QUIC. Although a monitored network flow might include other QUIC packets that contain encrypted "200 OK" confirmations, a quick series of such packets is a rare pattern.

ISP-side QoE inference. To identify this distinctive pattern in each monitored network flow, an ISP runs IQN's ISP agent on the data plane of its routers. The ISP agent identifies QUIC-encrypted *candidate packets* sized between 1,180 and 1,288 bytes, which might contain "200 OK" confirmations. Because some of these packets might be unrelated to an

IQN signal, the ISP agent has to discern and filter them out. Additionally, the client-server-agent path might experience packet loss, reordering, and significant jitter due to in-network and end-host processing [32], causing variations in the count and timing of packets representing an IQN signal upon its arrival at the ISP agent.

The algorithm for stall inference operates in two modes: *out-of-stall* and *in-stall*. In the out-of-stall mode, the ISP agent monitors packet arrival times by sliding a *stall-start window* over them. It infers the start of a new stall when the window contains at least n packet arrivals. The algorithm registers the first arrival time as start time s of the stall and switches to the in-stall mode. In the in-stall mode, it slides a *stall-end window* over packet arrival times and infers the end of the stall when the window contains fewer than n packet arrivals. The ISP agent records the first of these times as end time e of the stall, estimates the stall duration as $e - s$, and then switches back to the out-of-stall mode.

We keep the algorithm simple by sizing both stall-start and stall-end windows relative to signaling interval p, instead of introducing new independent parameters. We size the stall-start window to $w_s = \frac{3p}{2}$, i.e., 50% longer than interval p. Despite potential timing distortions along the delivery path, we expect this duration to be long enough for n packets to arrive and inform the ISP agent about the onset of a stall. On the other hand, $w_s = \frac{3p}{2}$ is short enough to avoid inferring a stall spuriously due to the arrival of unrelated candidate packets. We empirically set the stall-end window to $w_e = 4p$ to reduce false negatives, which might occur if the end-user agent fails to emit an IQN signal during an actual stall. Because unrelated candidate packets are infrequent, they do not interfere with inferring the end of a stall when w_e is as long as $4p$. Additionally, setting $w_e = 4p$ ensures that this window is shorter than the typical gaps between actual stalls, thus preventing the inference of multiple stalls as one.

Per-flow tracking of packet sizes and arrival times in IQN-assisted QoE inference imposes less overhead than the stateful packet processing required for traditional data-plane DPI, which advanced routers already support for independent ISP-side QoE inference and more complex tasks [33], [34]. Therefore, we expect the proposed method to scale to ISPs in the Internet core.

User-side QoE detection. To signal QoE, the end-user agent has to detect it first. One possibility is to leverage YouTube's "stats for nerds", a feature that provides end users with detailed QoE information. We demonstrate IQN's feasibility and effectiveness in more general settings, where an OTT provider does not disclose detailed QoE metrics and instead offers limited visual cues, such as the spinning icon in the center of YouTube's playback area during a stall, to indicate temporary QoE impairments.

YouStall's end-user agent detects stalls by periodically capturing a screenshot of the playback area at interval p. If the central part of two consecutive screenshots changes while the periphery remains the same, the end-user agent attributes the change to the spinning icon and infers a potential stall. To avoid sending spurious IQN signals, the end-user agent strives to minimize false positives by incorporating logic to filter out fictitious stall detections caused by user actions, such as clicking the progress bar. While the end-user agent does not detect stalls shorter than p, this feature aligns with YouStall's aim to inform ISPs about longer stalls that noticeably disrupt end-user QoE and are resolvable by ISPs.

ISP-side QoE improvement. We develop the YouStall prototype for IQN-assisted QoE inference primarily to evaluate the effectiveness of IQN signaling, which is our main innovation. QoE improvement via in-network techniques, such as flow prioritization and rerouting, is well-researched and offers well-known performance benefits. For example, [35] demonstrates that these techniques enable ISPs to effectively mitigate temporary QoE impairments, including stalls, when informed through out-of-band signaling. Enhancing YouStall with similar QoE-improving mechanisms is a direction for future work.

YouStall implementation. We implement YouStall in Python 3.10. YouStall's end-user agent utilizes Pillow [36] for image manipulation and sets the PAUSE parameter of PyAutoGUI [37] to 0 for rapid toggling of the autoplay switch. The ISP agent employs pyshark [38] for real-time packet capture and analysis. This paper openly releases its code and experimental configurations on GitHub [39].

IV. Evaluation

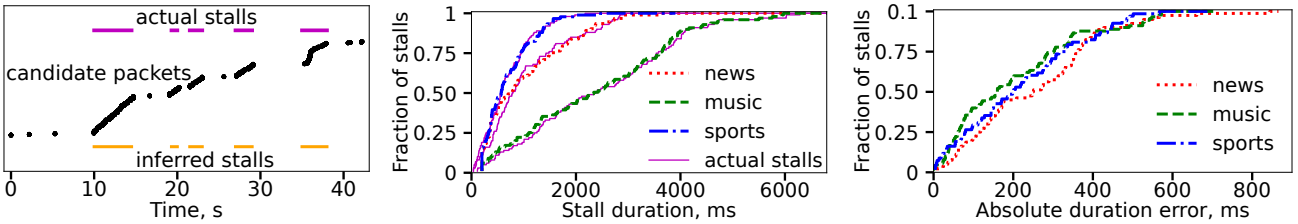
A. Experimental setup

IQN's end-user agent runs on a Madrid-based Intel i7 machine (six cores, 2.6 GHz CPU, 16 GB RAM) with Linux Ubuntu 22.04.4 LTS. Located in Frankfurt, approximately 1,500 km away, an EC2 instance (one core, 2.4 GHz Intel Xeon CPU, 1 GB RAM) with Linux Ubuntu 24.04.4 LTS emulates IQN's ISP agent. We tunnel the end-user device's Internet traffic to the EC2 instance using OpenVPN [40]. An automated script plays YouTube Live [29] video streams from the news, music, and sports genres on the end-user device through the Google Chrome browser.

Since YouTube stalls are typically infrequent, our evaluation induces 100 stalls per genre by fixing the video resolution at 720p and using tcconfig [41] to limit the YouTube traffic to 1 Mbps [42]. We collect ground truth on the stalls by employing Selenium [43] to track the ytp-spinner tag, which renders the spinning icon in YouTube's interface. We synchronize the timelines of the stalls and captured screenshots. To assess memory and CPU (central processing unit) overhead, we utilize psutil [44].

B. Experimental results

Figure 2a illustrates YouStall's operation. The horizontal magenta segments indicate actual stalls, which prompt IQN signaling from the end-user agent. Each black dot represents the arrival of a candidate packet at the ISP agent. Between stalls, the ISP agent observes infrequent candidate packet arrivals unrelated to IQN signaling. During a stall, the ISP agent detects a higher arrival rate of candidate packets and



(a) An operational example (b) Duration of inferred vs. actual stalls (c) Duration error for significant stalls

Fig. 2: IQN signaling from the end user enables accurate QoE inference by the ISP along YouTube's server-to-client path.

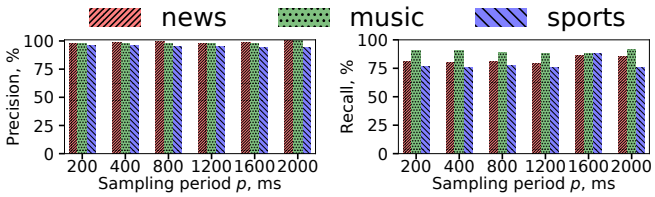


Fig. 3: Precision and recall of user-side QoE detection.

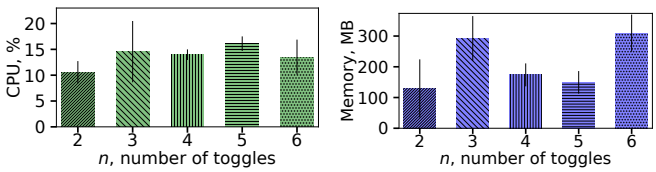


Fig. 4: Overhead of YouStall's end-user agent.

correctly infers the stall. Figure 2a depicts the inferred stalls as horizontal orange segments.

Figure 2b shows the inferred stall-duration distributions for the news, music, and sports genres, with the respective ground-truth distributions plotted as solid magenta lines. Although the actual stall-duration distributions for the three genres are distinct (with averages of 1.1, 2.6, and 0.7 s), the inferred and actual distributions for each genre are very similar, indicating that YouStall accurately infers stall-duration distributions.

Since we experiment with YouStall in real-world conditions, where the path between the two agents introduces substantial jitter of up to 1 s, evaluating the inference accuracy for each stall is challenging. We utilize the end-user and ISP-agent timelines to match actual and inferred stalls within a window of 1 s. For significant stalls lasting at least 400 ms, we confidently identify 238 one-to-one mappings between inferred and actual stalls across the three genres (from a total of 300 actual stalls). The average MAE and RMSE for these 238 mappings are 231 and 288 ms, respectively. Figure 2c depicts the absolute error of the inferred stall duration for each genre. These results suggest that YouStall estimates the duration of significant stalls with relatively low error.

Based on the synchronized timelines of actual stalls and captured screenshots, Figure 3 presents the precision and recall of YouStall's user-side QoE detection. The sampling interval has minimal impact on these metrics. The end-user agent detects stalls with nearly 100% precision, effectively avoiding spurious IQN signals. However, recall is lower because YouStall, by design, does not detect stalls shorter than p. Because shortening p increases the number of detected stalls, we set the default sampling interval to the shortest examined value, $p = 200$ ms.

Figure 4 presents the overhead of YouStall's end-user agent, based on four runs with 10 actual stalls. The CPU and memory consumption are approximately 12% and 200 MB, respectively. This overhead is manageable and does not significantly depend on the number of toggles constituting an IQN signal.

V. Related work

Independent ISP-side QoE inference. [4] identifies video segments in encrypted traffic by examining packet sizes and timing. Based on combinatorial matching, [5] infers video resolution when QUIC multiplexes video and audio content. [6]–[8] employ decision trees, random forests, and convolutional neural networks, respectively, to infer various QoE metrics. Other solutions for QoE detection from encrypted traffic include [9]–[11]. Despite the growing effectiveness of learning methods, independent QoE inference by an ISP still struggles to achieve high performance. In contrast, IQN enables accurate user-assisted ISP-side QoE inference.

ISP-OTT cooperation. Collaboration with OTT providers promises tangible benefits for ISPs. While [13] explores ISP-OTT cooperation in software-defined networking to jointly optimize intra-domain and inter-domain routing, [14] enables cooperation between OTT hypergiants and remote ISPs without requiring direct peering. Whereas practice shows that OTT providers are reluctant to assist ISPs in QoE inference, we design IQN to operate without any explicit assistance from the OTT provider.

QoE signaling from end users. While ALTO [18] and P4P [19] enable QoE signaling from the end user to an ISP, the out-of-band nature of these mechanisms is only weakly compatible with modern Internet realities, such as multi-ISP paths and NAT. In-band signaling via network cookies [17] differs from IQN by requiring support from the OTT server.

In-band notification. Similarly to IQN, ECN [30] is an in-band mechanism for conveying congestion-related information. Whereas ECN sends congestion signals from networks to end devices for transport-layer reactions and does not distinguish between applications, IQN enables an end-user agent

to notify networks about congestion-triggered application-specific QoE impairments. Unlike IQN, which functions as an emergency signal for transient QoE-impairing congestion, the in-band signaling in RD (rate-delay) network services [45] supports persistent differential treatment of application classes that prioritize either high throughput or low latency for their network flows.

User-side QoE detection. While VideoEye [46] detects QoE offline from recordings of the OTT client's screen, Tero [47] extracts QoE from thumbnails in gaming footage. In contrast, our YouStall prototype captures screenshots to detect and signal QoE in real time. While [48] evaluates how user interactions with an OTT client affect QoE of encrypted video sessions, YouStall leverages the OTT client's interface to induce a distinctive pattern of packet transmission from the OTT server.

ISP-side QoE improvement. ISP-side methods for QoE management include traffic prioritization [1] and rerouting [2]. We envision IQN operating in conjunction with similar application-aware traffic engineering and prioritization techniques, rather than relying on more static approaches to traffic management, such as the MED (multi-exit discriminator) attribute in BGP (border gateway protocol) [49].

VI. Discussion and conclusion

Deployment and robustness. We expect IQN's end-user and ISP agents being developed and maintained by an ISP, an ISP consortium, or a third party. This maintenance would involve updating the agents in response to changes made by the OTT provider to its client's interface.

Interference by the OTT provider. Beyond simply not assisting, OTT providers might deliberately disrupt IQN signaling. Such actions would frustrate IQN-agent developers and ISPs, potentially leading to the introduction of neutrality regulations for OTT providers, similar to those that currently govern ISPs.

OTT-specific IQN instantiation. OTT services differ significantly in their interface features and operation, potentially requiring a tailored IQN instance for each service. Since the number of OTT hypergiants is relatively small, maintaining separate IQN instances for all of them appears scalable.

Future work. The primary contribution of this paper is IQN signaling, which enables accurate ISP-side QoE inference from encrypted traffic without any support from OTT providers. While our preliminary results substantiate IQN's promise, this pioneering work opens up numerous directions for further research, such as encoding additional QoE factors into IQN signals and improving QoE inference from observed packet patterns. We also plan to enhance the YouStall prototype with QoE-impairment mitigation mechanisms and evaluate their performance gains in real large-scale network environments.

Acknowledgments

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

References

[1] A. Nikravesh, D. K. Hong, Q. A. Chen, H. V. Madhyastha, and Z. M. Mao, "QoE Inference Without Application Control," in *Internet-QoE*, 2016.

[2] M. Apostolaki, A. Singla, and L. Vanbever, "Performance-Driven Internet Path Selection," in *SOSR*, 2021.

[3] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W.-T. Chang, and Z. Shi, "The QUIC Transport Protocol: Design and Internet-Scale Deployment," in *SIGCOMM*, 2017.

[4] S. Xu, S. Sen, and Z. M. Mao, "CSI: Inferring Mobile ABR Video Adaptation Behavior Under HTTPS and QUIC," in *EuroSys*, 2020.

[5] Y. Zhao, H. Wu, L. Chen, S. Liu, G. Cheng, and X. Hu, "Identifying Video Resolution from Encrypted QUIC Streams in Segment-Combined Transmission Scenarios," in *NOSSDAV*, 2024.

[6] M. H. Mazhar and Z. Shafiq, "Real-Time Video Quality of Experience Monitoring for HTTPS and QUIC," in *INFOCOM*, 2018.

[7] C. Gutterman, K. Guo, S. Arora, T. Gilliland, X. Wang, L. Wu, E. Katz-Bassett, and G. Zussman, "Requet: Real-Time QoE Metric Detection for Encrypted YouTube Traffic," *ACM TOMM*, vol. 16, no. 2s, pp. 1–28, 2020.

[8] M. Shen, J. Zhang, K. Xu, L. Zhu, J. Liu, and X. Du, "DeepQoE: Real-Time Measurement of Video QoE from Encrypted Traffic with Deep Learning," in *IWQoS*, 2020.

[9] S. Wassermann, M. Seufert, P. Casas, L. Gang, and K. Li, "ViCrypt to the Rescue: Real-Time, Machine-Learning-Driven Video-QoE Monitoring for Encrypted Streaming Traffic," *IEEE TNSM*, vol. 17, no. 4, pp. 2007–2023, 2020.

[10] T. Sharma, T. Mangla, A. Gupta, J. Jiang, and N. Feamster, "Estimating WebRTC Video QoE Metrics Without Using Application Headers," in *IMC*, 2023.

[11] Tisa-Selma, A. Bentaleb, and S. Harous, "Inferring Quality of Experience for Adaptive Video Streaming over HTTPS and QUIC," in *IWCMC*, 2020.

[12] P. Gigis, M. Calder, L. Manassakis, G. Nomikos, V. Kotronis, X. Dimitropoulos, E. Katz-Bassett, and G. Smaragdakis, "Seven Years in the Life of Hypergiants' Off-Nets," in *SIGCOMM*, 2021.

[13] Y. Zhao, A. Saeed, M. H. Ammar, and E. W. Zegura, "Unison: Enabling Content Provider/ISP Collaboration Using a vSwitch Abstraction," in *ICNP*, 2019.

[14] C. Munteanu, O. Gasser, I. Poese, G. Smaragdakis, and A. Feldmann, "Enabling Multi-Hop ISP-Hypergiant Collaboration," in *ANRW*, 2023.

[15] V. Stocker, G. Smaragdakis, and W. Lehr, "The State of Network Neutrality Regulation," *SIGCOMM CCR*, vol. 50, no. 1, pp. 45–59, 2020.

[16] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet," *IEEE/ACM ToN*, vol. 13, no. 3, pp. 462–475, 2005.

[17] Y. Yiakoumis, S. Katti, and N. McKeown, "Neutral Net Neutrality," in *SIGCOMM*, 2016.

[18] S. Shalunov, R. Wendy, R. Woundy, S. Previdi, S. Kiesel, R. Alimi, R. Penno, and Y. R. Yang, "Application-Layer Traffic Optimization (ALTO) Protocol," IETF RFC 7285, 2014.

[19] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: Provider Portal for Applications," in *SIGCOMM*, 2008.

[20] L. Gao, "On Inferring Autonomous System Relationships in the Internet," *IEEE/ACM ToN*, vol. 9, no. 6, pp. 733–745, 2001.

[21] S. Hasan and S. Gorinsky, "Obscure Giants: Detecting the Provider-Free ASes," in *IFIP Networking*, 2012.

[22] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. Vazirani, "How Many Tiers? Pricing in the Internet Transit Market," in *SIGCOMM*, 2011.

[23] V. Giotsas, G. Smaragdakis, B. Huffaker, M. Luckie, and k. claffy, "Mapping Peering Interconnections to a Facility," in *CoNEXT*, 2015.

[24] I. Castro, J. C. Cardona, S. Gorinsky, and P. Francois, "Remote Peering: More Peering without Internet Flattening," in *CoNEXT*, 2014.

[25] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a Large European IXP," in *SIGCOMM*, 2012.

[26] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*. Morgan Kaufmann, 2021.

[27] C. Lan, J. Sherry, R. A. Popa, S. Ratnasamy, and Z. Liu, "Embark: Securely Outsourcing Middleboxes to the Cloud," in *NSDI*, 2016.

[28] Amazon Web Services, "Amazon EC2," https://aws.amazon.com/ec2, 2024.

[29] Google LLC YouTube, "YouTube Live," https://www.youtube.com/live, 2024.

[30] K. K. Ramakrishnan, S. Floyd, and D. L. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," IETF RFC 3168, 2001.

[31] R. Schatz, T. Hoßfeld, and P. Casas, "Passive YouTube QoE Monitoring for ISPs," in *IMIS*, 2012.

[32] V. Arun, M. Alizadeh, and H. Balakrishnan, "Starvation in End-to-End Congestion Control," in *SIGCOMM*, 2022.

[33] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano, "nDPI: Open-Source High-Speed Deep Packet Inspection," in *IWCMC*, 2014.

[34] J. Hypolite, J. Sonchack, S. Hershkop, N. Dautenhahn, A. DeHon, and J. M. Smith, "DeepMatch: Practical Deep Packet Inspection in the Data Plane Using Network Processors," in *CoNEXT*, 2020.

[35] T. Tran, D. Gageot, C. Neumann, G. Bichot, A. Tlili, K. Boutiba, and A. Ksentini, "On the Benefits and Caveats of Exploiting Quality on Demand Network APIs for Video Streaming," in *NOSSDAV*, 2024.

[36] J. A. Clark, "pillow 10.3.0," Python Sofware Foundation, https://pypi.org/project/pillow/, 2024.

[37] A. Sweigart, "Pyautogui 0.9.54," Python Software Foundation, https://pypi.org/project/PyAutoGUI/, 2024.

[38] KimiNewt, "pyshark 0.6," Python Software Foundation, https://pypi.org/project/pyshark/, 2024.

[39] L. Peroni, S. Gorinsky, and F. Tashtarian, "IQN and YouStall: Code and Experimental Configurations," GitHub, https://github.com/Leo-rojo/IQN_YouStall_Code_CloudNet_2024, 2024.

[40] OpenVPN Inc., "OpenVPN," https://openvpn.net/, 2024.

[41] T. Hombashi, "tcconfig 0.7.0a1," Python Sofware Foundation, https://pypi.org/project/tcconfig/0.7.0a1/, 2024.

[42] J. Liu, D. Lerner, J. Chung, U. Paul, A. Gupta, and E. Belding, "Watching Stars in Pixels: The Interplay of Traffic Shaping and YouTube Streaming QoE over GEO Satellite Networks," in *PAM*, 2024.

[43] Python Software Foundation, "Selenium 4.22.0," https://pypi.org/project/selenium/, 2024.

[44] G. Rodola, "psutil 6.0.0," Python Software Foundation, https://pypi.org/project/psutil/, 2024.

[45] M. Podlesny and S. Gorinsky, "Leveraging the Rate-Delay Trade-Off for Service Differentiation in Multi-Provider Networks," *IEEE JSAC*, vol. 29, no. 5, pp. 997–1008, 2011.

[46] S. Xu, E. Petajan, S. Sen, and Z. M. Mao, "What You See Is What You Get: Measure ABR Video Streaming QoE via On-Device Screen Recording," in *NOSSDAV*, 2020.

[47] C. Alvarez and K. Argyraki, "Learning a QoE Metric from Social Media and Gaming Footage," in *HotNets*, 2023.

[48] I. Bartolec, "Performance Estimation of Encrypted Video Streaming in Light of End-User Playback-Related Interactions," in *MMSys*, 2021.

[49] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," IETF RFC 4271, 2006.