Watching Stars in Pixels: The Interplay of Traffic Shaping and YouTube Streaming QoE over GEO Satellite Networks

Jiamo Liu[1], David Lerner[2], Jae Chung[2], Udit Paul[3] *, Arpit Gupta[1], and Elizabeth Belding[1]

[1] {jiamoliu,arpitgupta,ebelding@ucsb.edu}; University of California, Santa Barbara
[2] {david.lerner,jaewon.chung@viasat.edu}; Viasat
[3] {udit.paul@ookla.com}; Ookla

Abstract. Geosynchronous satellite (GEO) networks are an important Internet access option for users beyond terrestrial connectivity. However, unlike terrestrial networks, GEO networks exhibit high latency and deploy TCP proxies and traffic shapers. The deployment of proxies effectively mitigates the impact of high network latency in GEO networks, while traffic shapers help realize customer-controlled data-saver options that optimize data usage. However, it is unclear how the interplay between GEO networks' high latency, TCP proxies, and traffic-shaping policies affects the quality of experience for commonly used video applications. To address this gap, we analyze the quality of over 2 k YouTube video sessions streamed across a production GEO network with a 900 Kbps shaping rate. Given the average bit rates of the videos, we expected streaming to be seamless at resolutions of 360p, and nearly seamless at resolutions approaching 480p. However, our analysis reveals that this is not the case: 30% of both TCP and QUIC sessions experience rebuffering, while the median average resolution is only 404p for TCP and 360p for QUIC. Our analysis identifies two key factors that contribute to sub-optimal performance: (i) unlike TCP, QUIC only utilizes 70% of the network capacity; and (ii) YouTube's chunk request pipelining neglects network latency, resulting in idle periods that disproportionately harm the throughput of smaller chunks. As a result of our study, Viasat discontinued support for the low-bandwidth data-saving option in U.S. business and residential markets to avoid potential degradation of video quality—highlighting the practical significance of our findings.

Keywords: Video streaming · Geosynchronous satellite network · Quality of experience · Quality of service.

1 Introduction

Geosynchronous satellite (GEO) networks, through providers such as Viasat and HughesNet, are a key last-mile Internet access technology in challenging environments such as rural and other hard-to-reach communities, aircraft, sea ships,

* Udit Paul was a PhD student at the University of California, Santa Barbara when the work was performed.

and others. While recently low Earth orbit (LEO) satellite networks have grown in availability, the high cost of deploying and maintaining LEO networks can hinder access for under-served communities [28], making GEO networks an attractive alternative. GEO satellites orbit 22 k miles above the earth and move at the same speed as the earth, ensuring that their location remains fixed relative to the ground stations over time. While convenient for routing simplification, the geosynchronicity comes at the cost of high latency; round trip times through GEO satellites are typically 500-600 ms. To mitigate the effects of this latency, GEO ISPs often employ a variety of techniques. These usually include TCP Performance Enhanced Proxies (PEP) with geo-optimized configurations. Additionally, many wireless plans, both terrestrial and satellite, provide customers with a fixed high-speed data quota (gigabytes) per month. After a customer consumes this data, their traffic is deprioritized, which can result in slow speeds for the user when the network is congested. To avoid data deprioritization, customers are typically provided with an option to reduce general data consumption by shaping their video traffic, thereby enabling high-speed data to last longer.

As the dominant Internet application, accounting for approximately 53% [6] of total Internet traffic, video streaming is a critical application to support in all network types. Particularly in remote areas, video streaming can be critical for activities such as online education and work. Prior studies of video streaming QoE have typically been conducted either in emulated or production terrestrial networks characterized by low delay and high bandwidth (e.g. [32, 12]). Amongst the results, these studies show that QoE degradation events are rare in terrestrial networks. Prior studies have also analyzed the impact of the transport layer in video performance, in particular comparing TCP and QUIC, the latter of which has gained wide adoption by applications such as YouTube [7, 4, 31, 29]. Some studies have shown that TCP and QUIC have similar performance for video streaming tasks in terrestrial networks [4, 31, 20], while in [14] Google claims that QUIC improves YouTube QoE key performance indicators (KPIs). However, because of their inherent differences, it is not clear whether these results hold in GEO networks, particularly when traffic shaping is employed.

We address this understanding gap through an extensive study of video streaming performance over an operational GEO satellite network. We focus our study on YouTube because of its widespread popularity; current data places YouTube video consumption as outpacing that of Netflix worldwide.[4] We stream 2,080 180-190 second videos, using either the TCP or QUIC protocol, and analyze the resulting QoE KPIs to characterize the video performance. Given the video bit rates to support 360p and 480p resolution, we expect consistent seamless streaming at an average resolution exceeding 360p. Critically and surprisingly, our first key discovery is that neither TCP nor QUIC achieves consistent seamless streaming at an average resolution of more than 360p when traffic is shaped at 900 Kbps, the rate offered to customers in our production network. Specifically,

[4] For instance, one study states that, in 2022, YouTube represented 15% of traffic on consumer broadband networks, while Netflix represented 9% [6].

we observe that 30% of each of TCP and QUIC sessions experience rebuffering events, while the median average resolution is 404p for TCP and 360p for QUIC.

To understand this observation, we analyze the video traffic and determine that TCP utilizes all available link capacity during transmission, while QUIC is only able to utilize 70% of the capacity. Our results suggest that the performance of QUIC, specifically in conjunction with the BBR congestion control algorithm, is suboptimal when used in GEO networks. Furthermore, both TCP and QUIC suffer from imperfect chunk request scheduling, resulting in idle time that further reduces the overall throughput for TCP by 36% and QUIC by 26%.

As a result of our study, Viasat discontinued support for the low-bandwidth data-saving shaping option in its U.S. business and residential network to avoid potential degradation of video quality. We encourage YouTube and other content providers to more fully consider the operational environment of GEO networks and optimize players to deliver high quality video despite the presence of high latency links and other GEO network features. Additionally, optimization is needed for the QUIC + BBR [14] stack to achieve performance comparable to PEP-enabled TCP + BBR in GEO satellite networks.

2 Background and Motivation

In this section, we provide background on the three key concepts in this paper. First, we describe video streaming, and in particular the use of adaptive bit rate algorithms and the KPIs that are used to measure video QoE. The characteristics of GEO satellite networks are then discussed, along with optimizations incorporated to provide customers with improved performance. Finally, we describe key features of the QUIC protocol, including a discussion of why QUIC is not necessarily the better protocol for GEO satellite networks despite its apparent suitability for video streaming.

2.1 Video streaming applications (VSAs)

Internet video streaming services typically divide a video into smaller segments called chunks. These chunks are often of different playback durations; hence the chunks can be of variable size [22]. The video quality is determined by the number of pixels in each frame (i.e., resolution) and the (average) number of bits per second of playback (i.e., bit rate). Most streaming service providers use variable bit rates (VBR) to encode the video into a sequence of frames. The number of bits needed to encode a specific chunk depends on the video type and its quality [17]. In general, high-action/high-resolution chunks require more bits to encode than motionless and/or lower-resolution chunks.

Adaptive bit rate algorithms. To optimize the viewing experience, each client maintains a buffer where it stores received chunks. With this repository, the likelihood of continuous playback during a video session is greatly increased. At the video session start, the client waits to fill this buffer to a predefined level before video playback begins. The client begins by sending an HTTPS request to retrieve a specific segment at a pre-selected quality (e.g., 360p). On receiving this request, the server sends the requested segment to the client.

Each client uses an application-layer *adaptive bit rate (ABR)* algorithm to determine the quality of the request in the next segment. The ABR algorithms employed by most video streaming services are proprietary, but previous work has shown that these algorithms typically use parameters such as estimated bandwidth and current buffer size to determine the quality of the next requested segment [38, 18].

Quality of experience for VSAs. Video stream QoE is determined by several KPIs. These include initial buffering time, resolution, and the number of rebuffering events. During a rebuffering event, video playback is paused while the received video is placed into the playback buffer. For optimal QoE, rebuffering events, resolution switches, and initial buffering time should be minimized [30]. A higher resolution is preferred as long as there is sufficient network bandwidth to deliver the video chunks before the playout deadline [11].

The goal of the ABR algorithm is to select the appropriate resolution for each chunk to maximize viewing resolution while also minimizing events that lower QoE. Most video streaming applications work very well in high-bandwidth (more than 10-15 Mbps), low-latency (few tens of ms) networks [26]. However, it is far less clear how well they perform in high-latency networks and shaped bandwidth, which are typical of GEO networks. Further, it is not well-understood how well these applications are able to interact with additional network components, such as TCP proxies and traffic-shaping algorithms, that are common in GEO networks. Hence, it is in these environments that our work focuses.

2.2 Geosynchronous satellite networks

Geosynchronous satellite networks use wireless links between ground stations and space satellites to connect subscribers to the Internet. Transparent TCP proxies, which speed up TCP slow-start and congestion recovery, are often employed to mitigate the effects of the long round-trip propagation delays of GEO links. Two TCP proxies are typically used: upstream and downstream. The upstream proxy runs at the ground station, while the downstream proxy runs at the satellite modems closer to the end-users. As a result of these proxies, each video streaming session entails three independent TCP connections: server-to-upstream-proxy (C1), upstream-to-downstream-proxy (C2), and downstream-proxy-to-client (C3), as shown in Figure 1. The upstream proxy acknowledges packets coming from the video server aggressively and therefore increases the congestion window quickly. On the client side, packets are acknowledged immediately as well.

In addition to TCP proxies, many satellite and mobile wireless network operators provide data-saver options that use traffic shapers to constrain the bandwidth allocated to different applications [15]. This enables users to view more hours of video or engage in other network activities while reducing the likelihood they exceed their monthly data limit.

QUIC in GEO satellite networks: QUIC is rapidly becoming the default transport layer protocol for many online services [14, 19]. YouTube, for example, is predominantly transmitted over QUIC. QUIC runs in user space, uses UDP

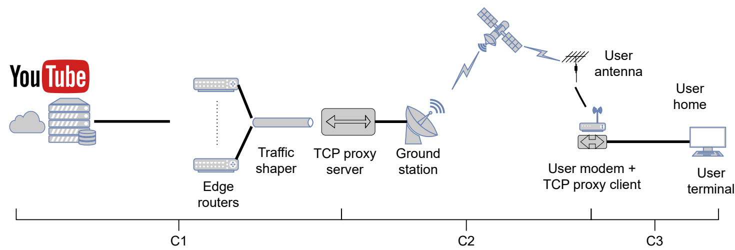

Fig. 1: Testbed configuration.

for data transmission, and is designed for applications that use multiple HTTPS multiplexed connections. Compared to TCP, it expedites the connection setup by reducing the number of round trips required to establish a connection between the two end hosts. During packet losses, QUIC avoids head-of-line blocking by servicing unaffected streams while recovering lost packets. TCP proxies cannot be leveraged by QUIC-based connections because QUIC connections are cryptographically signed end-to-end. Hence the feedback loop of QUIC is delayed in GEO networks. As a result of these key differences, it is unclear how QUIC will perform over GEO links and how video QoE will be impacted when used with other components such as the PEP and traffic shaper. This observation serves as the driving force behind our investigation.

3 Methodology and Dataset

In this section, we describe the configuration of our testbed and the methodology used to stream YouTube videos while collecting HTTP logs and QoE KPIs. Additionally, we provide a summary of the dataset and experiment metrics collected.

Testbed. Figure 1 illustrates the primary components of our testbed network architecture, including the YouTube server, traffic shaper, server-side TCP proxy, satellite link, client-side proxy, and client laptop. Note that because this is a production network, the client laptop is the only component over which we have direct control. To improve TCP performance over the long latency satellite link, TCP traffic is split into three separate connections (C1, C2, and C3), as shown in the figure. On the other hand, QUIC traffic is not split into separate connections due to its use of end-to-end encryption; the proxies simply forward the traffic. The client laptop is used to collect HTTP logs. The satellite provider uses a token bucket traffic shaper to limit the throughput of video traffic on the low-latency link between the YouTube server and the upstream proxy, with an average bandwidth shaping rate of 0.9 Mbps and bursts up to 0.99 Mbps. The authors of [15] found that multiple GEO satellite network ISPs utilize this shaping rate.[5] Importantly, we note that the playback resolution under either constant or variable bandwidth shaping will differ depending on the operation

[5] Note that traffic shaping is a subscriber opt-in feature for the ISP in the study.

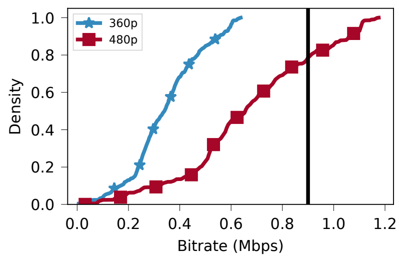

Fig. 2: Average video bit rates for each resolution. Vertical line is 900 Kbps, the shaped rate.

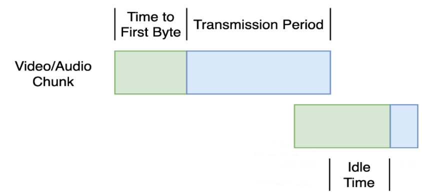

Fig. 3: Chunk request model.

of the ABR and VBR algorithms. The client side congestion control algorithm is CUBIC (Ubuntu Default); however, we are more interested in the server side congestion control (CC) algorithm because it determines how quickly video/audio data is sent to the client. The server side congestion control algorithm is BBRv3 [1]. Finally, the congestion control algorithm used between the client and server proxies is a proprietary version of ECN-enabled CUBIC with modifications such as a large initial congestion window. Upon detecting congestion, the window size is reduced by 1/3. The congestion window then increases slowly at first, and eventually grows exponentially, similar to TCP CUBIC. The technical details of the testbed are summarized in Table 1.

Operating System	Ubuntu 22.04
Browser	Google Chrome
Client Congestion Control	CUBIC with hystart
Server Congestion Control	BBRv3

Table 1: Technical details of the testbed.

YouTube experiments. We analyzed an open video catalog of approximately 8 million entries released by YouTube [2] to collect representative data from a variety of video types. We randomly selected 13 videos from each of the 16 distinct video categories[6] to form a total pool of 208 videos. We streamed each video with YouTube's production ABR ten times: five with QUIC enabled and five with QUIC disabled, for a total of 2,080 sessions; TCP was utilized in the QUIC-disabled sessions. We restarted the browser before each session to avoid caching. Each video is between 180-190 seconds in length to improve the scal-

[6] Categories collected are: Sports, Education, Science & Technology, Shows, Pets & Animals, Nonprofits & Activism, News & Politics, Gaming, Music, Comedy, People & Blogs, Autos & Vehicles, Film & Animation, Entertainment, Howto & Style, Travel & Events. Categories such as Sports are usually of higher bit rate compared to Education.

ability of the experiments while simultaneously being long enough to allow the congestion window to saturate at its maximum capacity [13]. The average bit rate of each of the 208 videos at 360p and 480p is represented as a CDF in Figure 2. The figure demonstrates that the videos selected are close to a uniform distribution; we verify this result for the other video resolutions but omit those results from the graph for clarity. The data collection process took place in October 2023.

Collection methodology. We assess the QoE of each video stream by gathering multiple well-defined QoE KPIs [30] for every YouTube session. Each experiment starts by randomly selecting a video and streaming it twice, once with QUIC enabled (we confirm that QUIC is used via HTTP logs) and once with it disabled (TCP enabled). The order of the QUIC/TCP protocols is reversed for each randomly chosen video to minimize bias introduced by CDN caching [3]. The resolution is set to automatic and a Chrome extension is utilized to capture player events at 250 ms intervals. The player events are then transmitted to a local server for storage, while a Python script driving Selenium is used to capture HTTP logs. This process is repeated until every video in the dataset is streamed five times with each transport protocol. We identify advertisement videos by timestamp and their different Video ID in "stats for nerds" and remove them from QoE and chunk analysis. The collected QoE KPIs are as follows:

- **Average session resolution**: given n available resolutions $R = \{R_1, R_2, \ldots R_n\}$ in a session, the fraction of time each resolution is viewed within the session is $P = \{P_1, P_2 \ldots P_n\}$; the average session resolution is given by $\sum_{i=1}^{n} P_i R_i$.
- **Initial buffering time**: the time from the instant when the video player connects to a server to the time when the first video frame is rendered and played. For this metric only, we exclude video sessions with pre-roll advertisements, and analyze 509 TCP and 509 QUIC sessions.
- **Resolution changes**: the number of resolution switches per video. Frequent switches usually lead to unsatisfactory user experience [30].
- **Rebuffering events**: the number of times the playback pauses due to insufficient buffered video. Few or no rebuffering events are desired for better QoE.

Video/audio chunks. Video and audio chunks are the fundamental operational unit of ABR algorithms; therefore, our analysis focuses on network performance at the chunk granularity. To obtain the performance of these chunks, we first filter out all HTTP `Network.requestWillBeSent` events reported by Chrome. Then we ensure these requests contain `?videoplayback` in their URL as well as `video` or `audio` as their MIME type. We keep track of the `request_id` of these events and obtain all `Network.dataReceived` events of the corresponding `request_id`. Finally, we group `Network.dataReceived` events by `request_id` to compute the size and performance metrics of each video/audio chunk.

Although such methods can only be applied if we have direct control over the client Chrome browser, previous work [9] has proposed methods to heuristically infer video/audio chunks based on the amount of data received between two

HTTP requests. However, based on our HTTP logs, we found that the heuristic approach may no longer be viable because 40% of video and audio chunks are smaller than the previously defined threshold of 80 KB; these chunks could be as small as 4 KB. Additionally, contrary to previous literature [9], new chunks can be requested before the completion of the previous chunk's transmission in GEO satellite networks. This requires us to model the chunk-downloading mechanism slightly differently as shown in Figure 3.

Chunk-level metrics. In a GEO network, video and audio chunks are primarily requested sequentially, but there are instances where a new request for additional chunks may be initiated before the previously requested chunks are fully received. To analyze the streaming behavior of the chunks, we utilize the following chunk-level metrics:

- **Chunk time to first byte (TTFB)**: the interval between the time a request is initiated and the first byte of data for the chunk is received. [25] find that this metric has an impact on chunk throughput; however, this should be differentiated from the idle time since TTFB does not consider previous chunks.
- **Idle time**: the interval between the end of the previous chunk's transmission and the beginning of the following chunk's transmission. Due to a large accumulated playback buffer, YouTube may occasionally decide to pause chunk requests for an extended period of time (tens of seconds). This causes the chunk throughput to be low irrespective of network conditions. Therefore, we only consider chunks that are requested within one second of the completion of the previous chunk. We also group chunks with negative idle time into larger chunks during our throughput analysis. This approach allows us to account for the overlapping download periods of these chunks.
- **Chunk download time**: the interval between when the first and final bytes of a chunk are received.
- **Chunk size**: the amount of data received in the requested chunk.
- **Throughput with idle time** (T_{idle}): the chunk throughput when idle time is taken into account. This metric demonstrates how imperfect request pipelining affects the throughput of chunks. Under shaped bandwidth, where buffer health remains relatively low, video chunk requests are made continuously and in a sequential manner. When these requests are perfectly pipelined, there should not be any idle time. The metric is given by:
  $T_{idle} = \frac{Chunk\_Size}{Idle\_Time + Chunk\_Download\_Time}$
- **Throughput without idle time** ($T_{network}$): the chunk throughput when idle time is not taken into consideration. This demonstrates how effectively the chunk download is using the underlying network resource and is given by:
  $T_{network} = \frac{Chunk\_Size}{Chunk\_Download\_Time}$

4 Video Stream Performance

A key goal of our study is to understand the YouTube QoE received by users when they enable video bandwidth shaping to reduce data usage. As part of

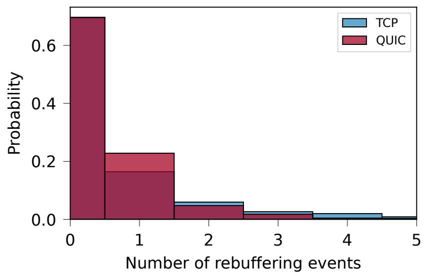

Fig. 4: Number of rebuffering events.

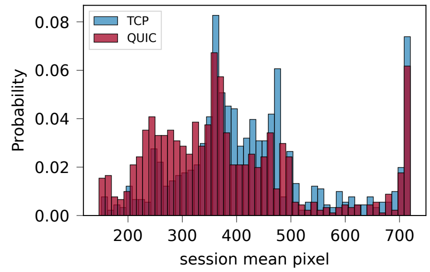

Fig. 5: Session average vertical pixel height.

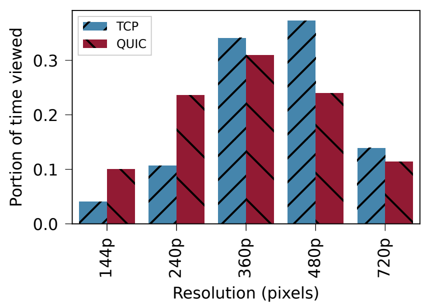

Fig. 6: Percent of time viewed at each resolution.

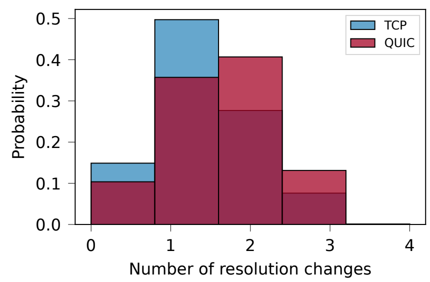

Fig. 7: Number of resolution changes.

this analysis, we quantify any performance differences based on the use of QUIC or TCP + PEP at a bandwidth shaping rate and a burst rate of 0.9 Mbps and 0.99 Mbps, respectively, which is the bandwidth shaping option commonly supported by GEO network providers, including in our Viasat production network [15].

We begin by analyzing the QoE KPIs. In Figure 2, we observed that the maximum average bit rate of 360p videos is only 0.63 Mbps, which is approximately 70% of the traffic shaping rate. Similarly, 0.9 Mbps is also more than the average bit rate of 70% of the videos at 480p. Consequently, we expect the average resolution to be more than 360p with minimal rebuffering events because of the ABR behavior. The ABR should stream at 480p and then switch to 360p or lower only when the buffer health becomes low. Hence rebuffering events should be minimal if the underlying network resource is utilized efficiently. However, contrary to our expectations, we find that only 70% each of the TCP and QUIC sessions do not experience rebuffering, as shown in Figure 4. Notably, the resolution achieved by QUIC sessions is significantly lower than that of TCP sessions, as shown in Figure 5. The median average session resolution for TCP is 404p, whereas for QUIC the median average session resolution is only 360p. Specifi-

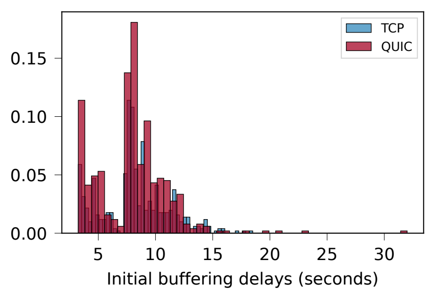

Fig. 8: Initial buffering time.

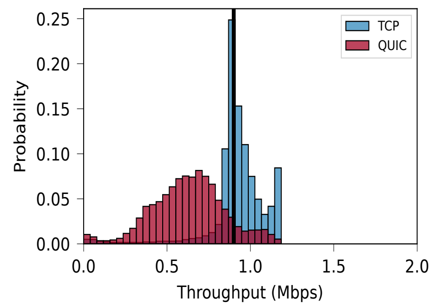

Fig. 9: Throughput without considering idle time. Vertical line is 900 Kbps.

cally, Figure 6 shows that 14% of the TCP streaming time is at a resolution less than 360p, while 33% of the QUIC streaming time is less than 360p. Surprisingly, these metrics indicate that neither the QUIC nor TCP sessions consistently meet our QoE expectations, with QUIC, in particular, struggling to stream at higher resolution consistently.

We next examine the initial buffering delay and number of resolution changes. Figure 7 shows that the number of resolution changes for TCP and QUIC are similar, with mean values of 1.28 and 1.57 per session, respectively. The mean initial buffering delay of TCP is 8.25 seconds while QUIC is 7.95 seconds, shown in Figure 8.

In summary, the results show that QUIC and TCP have similar numbers of rebuffering events, resolution changes and initial buffering delay. Further, neither QUIC nor TCP is able to stream seamlessly at an average resolution of more than 360p when bandwidth shaping is utilized. We hypothesize that there are suboptimal transport and/or application layer operations that manifest in unexpectedly poor performance in long-latency networks, preventing both TCP and QUIC, in particular, from fully utilizing available network resources and achieving better performance. In the following section, we explore chunk-level QoS to uncover potential reasons for lower-than-expected QoE.

5 Diagnosis of Sub-optimal QoE

Section 4 showed that both TCP and QUIC fail to deliver expected QoE KPIs, with QUIC, in particular, struggling to provide higher resolution. In this section, we dig deeper into these results to understand this performance anomaly.

5.1 Insufficient network resource utilization

Figures 2 and 6 demonstrated that sufficient bandwidth was available to stream at 360p, yet 33% of the time, QUIC streamed at a lower resolution. Our goal is to understand why QUIC is unable to utilize available network resources efficiently.

We begin with the throughput of individual chunks when the idle time is not considered, depicted in Figure 9. The figure shows that TCP clearly forms a peak around 900 Kbps, the shaped bandwidth, whereas QUIC's throughput widely varies well below 900 Kbps. The mean and median of TCP $T_{network}$ are 0.92 Mbps and 0.91 Mbps, respectively, while the mean and median of QUIC $T_{network}$ are 0.63 and 0.64 Mbps.

Both TCP and QUIC use BBR as their server-side congestion control algorithm. BBR continuously probes for the bottleneck bandwidth and propagation delay to determine the sending rate; however, we have seen that QUIC does not send at a rate that saturates the capacity of the link. Therefore, we hypothesize that the BBR algorithm performs differently with QUIC compared to TCP in GEO networks, likely as a result of PEP in the network architecture. Specifically, QUIC BBR experiences the full 600 ms of latency across the satellite link, whereas TCP BBR experiences only the low-latency link (typically less than 100 ms round trip time) between the YouTube server and PEP server proxy. The difference in this propagation delay may cause the YouTube server to incorrectly estimate the available bandwidth and send data at a rate that never fully saturates the link when QUIC is used. Prior work, such as [35] and [5], has also observed that the combination of TCP+PEP+BBR achieves higher throughput than QUIC+BBR. Relatedly, [35] showed that the slow-start phase of QUIC+BBR is the cause of the lower throughput; QUIC+BBR requires up to 10 seconds to reach maximum throughput, whereas TCP takes less than one second. In our analysis, we observe that $T_{network}$ of QUIC is still variable well below 900 Kbps even for video chunks requested 30 seconds or more after the session start (this result is included in the Appendix). As a result, the start-up phase is not the only factor that affects the performance of QUIC+BBR in our test network.

5.2 YouTube scheduling inefficiencies

Prior work has shown that many YouTube videos are not watched to completion [24]. As a result, YouTube tries to reduce unnecessary bandwidth consumption by limiting the number of chunks downloaded in advance. Our analysis has shown that it is possible for the currently requested chunk to start transmission before the completion of the previous chunk. Ideally, the YouTube ABR should learn the RTT of the network and request the next chunk slightly before the completion of the current chunk (the TTFB of each chunk is shown in the Appendix). This would minimize the amount of time spent idling by the network and therefore increase network utilization. However, as shown in Figure 10, we observe that more than 78% of the chunks experience idle time with both TCP and QUIC. Interestingly, we can also observe that a peak occurs around 600 ms. This suggests that the high propagation delay of the GEO network is not considered when making chunk requests. Note that we ignore chunks with long idle time caused by abundant playback buffers in our analysis. This allows us to focus on periods when high network utilization is critical.

To assess the impact of this idle time on smaller chunks whose throughput is dominated by idle time, we analyze the relationship between the size of the

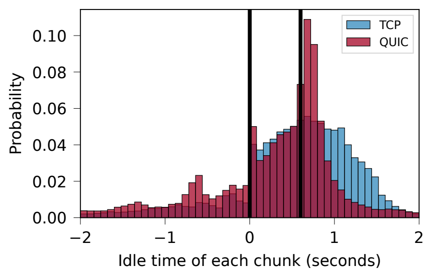

Fig. 10: Idle time. Vertical lines are 0 ms and 600 ms idle time.

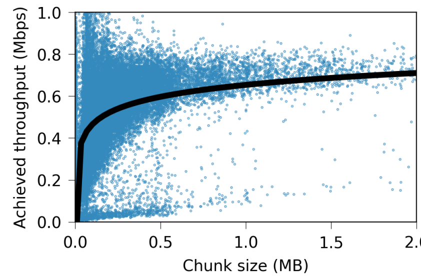

Fig. 11: Chunk size vs throughput for QUIC.

chunk downloaded and the throughput achieved for both TCP and QUIC. Figure 11 shows a logarithmic relationship between the two variables. Therefore, we calculate the Pearson correlation of achieved throughput and log(chunk size) to characterize the relationship. QUIC shows a positive Pearson correlation of 0.59 ($\ll 0.05$ p-value), whereas these two variables are also correlated for TCP with a Pearson statistic of 0.62 ($\ll 0.05$ p-value). Similar trends are observed in [16] for 4G and WiFi networks. This leads us to infer that these pipelining inefficiencies significantly reduce the throughput of smaller chunks. In addition, shaped traffic on GEO networks is disproportionately impacted due to their lower bandwidth and higher latency. According to [22], YouTube tends to request smaller chunks when bandwidth is lower. We observe that more than 53% of all chunks in our sessions are less than 0.1 MB; this, in turn, further exacerbates the problem. It is worth noting that the chunk-requesting inefficiency observed in our study may be specific to low-bandwidth high-latency scenarios.

To investigate this hypothesis, we compare the impact of idle time in the GEO network with that on our campus network, which is both high bandwidth and low latency. We stream the same 2,080 videos to a desktop in our campus research lab and find that the median idle time for both TCP and QUIC is short, approximately 15 ms (supporting the assumption of almost ideal sequential chunk requesting). Moreover, given the abundance of campus network capacity, the idle time has no practical impact on QoE in this setting, with achieved throughput T_{idle} exceeding 60 Mbps. In contrast, in the shaped GEO network, the median achieved throughput T_{idle} for TCP and QUIC is approximately 0.58 Mbps (0.64 $T_{network}$) and 0.47 Mbps (0.73 $T_{network}$), respectively. This suggests a correlation between idle time and the round trip propagation delay of the network, providing further evidence that the existing pipelining mechanism is sub-optimal in GEO networks. In summary, TCP experiences a 36% throughput reduction due to idle time, even with the link being saturated during transmission, while QUIC suffers a 27% reduction in throughput due to idle time, and then an additional 30% reduction because QUIC is unable to fully utilize the link capacity during transmission.

6 Related Work

Prior literature has studied the QoE of ABR streaming and YouTube [36, 23, 10, 21]. However, far fewer studies have investigated video streaming QoE in operational GEO networks. In [8], a framework that facilitates live 4k video streaming over a 5G core network via a GEO satellite backhaul is proposed. However, the focus of this work is live video streaming, as opposed to non-real-time video streaming in our study. [34] proposes a caching framework that improves video streaming QoE within GEO satellite networks. The limitations of this study include the utilization of an academic DASH player and the investigation of only a single video. Distorted versions of videos are generated in [37] by adjusting QoS parameters such as packet loss. The distorted videos are replayed to volunteers, and the corresponding Mean Opinion Score is recorded. This work uses an emulated satellite link, and the impact of ABR is not considered. A recent study [27] benchmarked multiple satellite ISPs across various tasks, including video streaming. The results are complementary to those of our study. Netflix's recent paper [33] demonstrates that server-side pacing effectively reduces congestion from bursty on-demand video transmission, maintaining QoE. This approach can also eliminate the need for ISPs to deploy traffic shaping. Extensive prior work has analyzed transport protocol performance [31, 4, 20]; however, these studies utilize a low-latency link. For instance, [31] uses a network emulator to shape home and mobile networks to 1 Mbps to study YouTube streaming QoE. In this environment, the authors conclude there is no meaningful difference between TCP and QUIC performance. Through the use of an academic DASH player in a terrestrial network and only one video, [4] concludes that the QUIC protocol does not improve QoE. Finally, the page load time difference between TCP and QUIC is studied in [20].

7 Conclusion

Our study characterizes the QoE KPIs of YouTube in a production GEO satellite network with 900 Kbps traffic shaping. We find that, despite the average bit rate of all the 360p resolution videos, and a majority of the 480p resolution videos, being less than the shaped bandwidth, the performance of these video streams is sub-optimal. Our work highlights the challenge of employing traffic shaping as an effective means to control video resolution, and therefore data-usage, on high latency networks, and the importance of accounting for network delay in chunk size and request timing for high quality video streaming. Many of the populations that stand to gain Internet access over the coming years will do so through non-traditional network architectures, including GEO networks. Application designers and content providers must consider a wide variety of network types and characteristics in their product, protocol design and content provisioning strategies to avoid unanticipated performance anomalies.

A Appendix

A.1 Ethical Considerations

Although our work involves HTTP log analysis on an operational GEO satellite network, our work is not human subjects research. At no point is any data collected from the customers of the network. We collect and analyze only our own experimentally generated traffic.

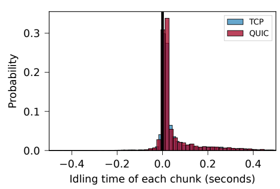

Fig. 12: Idle time of campus network.

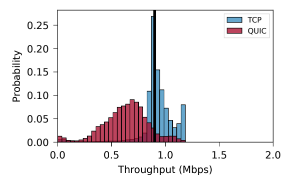

Fig. 13: Post 30 seconds throughput without considering idle time. Vertical line is 900 Kbps.

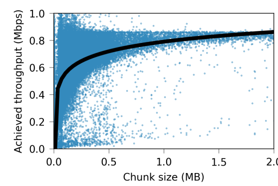

Fig. 14: Chunk size vs throughput over TCP.

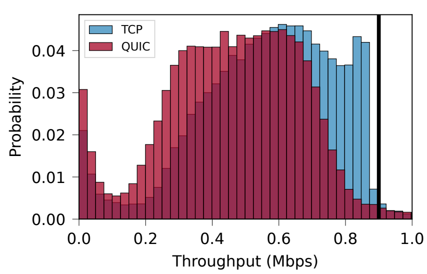

Fig. 15: Achieved throughput. Vertical line is 900 Kbps.

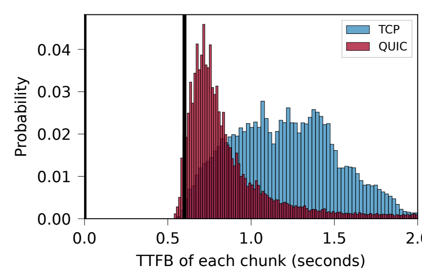

Fig. 16: Time to first byte of each chunk. Vertical line is 600 ms.

A.2 Supplementary Results

In this section we include some additional, supplementary graphs that were briefly described in the main body of the paper. The median idle time for both TCP and QUIC was short in our campus network experiment, around 15 ms, as shown in Figure 12. This result suggests that the pipelining inefficiency is magnified by the high round trip time of the GEO satellite network. Figure 13 shows the $T_{network}$ after 30 seconds of playback, in order to eliminate any effect due to slow start. The figure indicates that QUIC throughput still varies well below the shaped bandwidth 900 kbps. This indicates that congestion control, and specifically the initial slow start, are not the source of the low throughput. Figure 16 shows the TTFB of each chunk. We can observe that almost all chunks have a TTFB larger than 600 ms; QUIC in particular forms a cluster close to 600 ms. The correlation between achieved throughput (T_{idle}) and chunk size for TCP is illustrated in Figure 14. The Pearson statistic for correlation of achieved throughput and log(chunk size) is 0.62. Finally, Figure 15 shows that the T_{idle} of TCP outperforms that of QUIC in GEO networks; the median TCP throughput is 0.58 Mbps, while QUIC's median throughput is 0.47 Mbps. Importantly, however, neither reach the shaped bandwidth rate. Figure 16 shows the TTFB of each chunk. The median chunk TTFB for TCP is 1.21 seconds, while it is 0.78 seconds for QUIC.

References

1. BBR development group. https://groups.google.com/g/bbr-dev, accessed: 2024-01-13
2. Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., Vijayanarasimhan, S.: Youtube-8M: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675 (2016)
3. Adhikari, V.K., Jain, S., Chen, Y., Zhang, Z.L.: Vivisecting YouTube: An active measurement study. In: IEEE INFOCOM'12 (2012)
4. Bhat, D., Rizk, A., Zink, M.: Not so QUIC: A performance study of DASH over QUIC. pp. 13–18. NOSSDAV'17 (2017)
5. Border, J., Shah, B., Su, C.J., Torres, R.: Evaluating QUIC's Performance Against Performance Enhancing Proxy over Satellite Link. In: IFIP Networking Conference. pp. 755–760 (2020)
6. Cantor, L.: The Global Internet Phenomena Report. Sandvine, Waterloo, ON, Canada, Tech. Rep (2022)
7. Flach, T., Papageorge, P., Terzis, A., Pedrosa, L., Cheng, Y., Karim, T., Katz-Bassett, E., Govindan, R.: An Internet-Wide Analysis of Traffic Policing. p. 468–482. ACM SIGCOMM'16 (2016)
8. Ge, C., Wang, N., Selinis, I., Cahill, J., Kavanagh, M., Liolis, K., Politis, C., Nunes, J., Evans, B., Rahulan, Y., Nouvel, N., Boutin, M., Desmauts, J., Arnal, F., Watts, S., Poziopoulou, G.: QoE-Assured Live Streaming via Satellite Backhaul in 5G Networks. IEEE Transactions on Broadcasting **65**, 381–391 (2019)
9. Gutterman, C., Guo, K., Arora, S., Wang, X., Wu, L., Katz-Bassett, E., Zussman, G.: Requet: Real-Time QoE Detection for Encrypted YouTube Traffic. p. 48–59. ACM MMSys '19 (2019)
10. Hoßfeld, T., Seufert, M., Hirth, M., Zinner, T., Tran-Gia, P., Schatz, R.: Quantification of YouTube QoE via Crowdsourcing. In: IEEE International Symposium on Multimedia. pp. 494–499 (2011)
11. Huang, T.Y., Ekanadham, C., Berglund, A.J., Li, Z.: Hindsight: Evaluate Video Bitrate Adaptation at Scale. p. 86–97. ACM MMSys '19 (2019)
12. Khokhar, M.J., Ehlinger, T., Barakat, C.: From Network Traffic Measurements to QoE for Internet Video. In: IFIP Networking Conference (2019)

13. Kuhn, N., Michel, F., Thomas, L., Dubois, E., Lochin, E.: QUIC: Opportunities and threats in SATCOM. In: ASMS/SPSC (2020)
14. Langley, A., Riddoch, A., Wilk, A., Vicente, A., Krasic, C., Zhang, D., Yang, F., Kouranov, F., Swett, I., Iyengar, J., Bailey, J., Dorfman, J., Roskind, J., Kulik, J., Westin, P., Tenneti, R., Shade, R., Hamilton, R., Vasiliev, V., Chang, W.T., Shi, Z.: The QUIC Transport Protocol: Design and Internet-Scale Deployment. p. 183–196. ACM SIGCOMM'17 (2017)
15. Li, F., Niaki, A.A., Choffnes, D., Gill, P., Mislove, A.: A Large-Scale Analysis of Deployed Traffic Differentiation Practices. p. 130–144. ACM SIGCOMM '19 (2019)
16. Lv, G., Wu, Q., Wang, W., Li, Z., Xie, G.: Lumos: Towards better video streaming QOE through accurate throughput prediction. In: IEEE INFOCOM'22. pp. 650–659 (2022)
17. Mansy, A., Ammar, M., Chandrashekar, J., Sheth, A.: Characterizing Client Behavior of Commercial Mobile Video Streaming Services. ACM MoViD'14 (2018)
18. Mao, H., Netravali, R., Alizadeh, M.: Neural Adaptive Video Streaming with Pensieve. p. 197–210. ACM SIGCOMM '17 (2017)
19. Matt Joras, Yang Chi: How Facebook is bringing QUIC to billions. https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions
20. Megyesi, P., Krämer, Z., Molnár, S.: How quick is QUIC? In: 2016 IEEE ICC (2016)
21. Mok, R.K., Chan, E.W., Luo, X., Chang, R.K.: Inferring the QoE of HTTP Video Streaming from User-Viewing Activities. p. 31–36. ACM W-MUST '11 (2011)
22. Mondal, A., Sengupta, S., Reddy, B.R., Koundinya, M., Govindarajan, C., De, P., Ganguly, N., Chakraborty, S.: Candid with YouTube: Adaptive Streaming Behavior and Implications on Data Consumption. p. 19–24. ACM NOSSDAV'17 (2017)
23. Nam, H., Kim, K.H., Calin, D., Schulzrinne, H.: YouSlow: A Performance Analysis Tool for Adaptive Bitrate Video Streaming. SIGCOMM Comput. Commun. Rev. **44**, 111–112 (2014)
24. Nam, H., Kim, K.H., Schulzrinne, H.: QoE matters more than QoS: Why people stop watching cat videos. In: IEEE INFOCOM 2016 (2016)
25. Nam, Y.S., Gao, J., Bothra, C., Ghabashneh, E., Rao, S., Ribeiro, B., Zhan, J., Zhang, H.: Xatu: Richer Neural Network Based Prediction for Video Streaming. Measurement and Analysis of Computing Systems **5** (2021)
26. Ramachandran, S., Gryta, T., Dapena, K., Thomas, P.: The truth about faster internet: It's not worth it. Wall Street Journal (2019), https://www.wsj.com/graphics/faster-internet-not-worth-it/
27. Raman, A., Varvello, M., Chang, H., Sastry, N., Zaki, Y.: Dissecting the performance of satellite network operators. ACM CoNEXT '23 (2023)
28. Reznik, S., Reut, D., Shustilova, M.: Comparison of geostationary and low-orbit "round dance" satellite communication systems. IOP Conference Series: Materials Science and Engineering **971** (2020)
29. Rüth, J., Wolsing, K., Wehrle, K., Hohlfeld, O.: Perceiving QUIC: Do Users Notice or Even Care? p. 144–150. ACM CoNEXT'19 (2019)
30. Seufert, M., Egger, S., Slanina, M., Zinner, T., Hoßfeld, T., Tran-Gia, P.: A Survey on Quality of Experience of HTTP Adaptive Streaming. IEEE Communications Surveys & Tutorials **17**, 469–492 (2015)
31. Seufert, M., Schatz, R., Wehner, N., Casas, P.: QUICker or not? an Empirical Analysis of QUIC vs TCP for Video Streaming QoE Provisioning. In: 2019 ICIN. pp. 7–12 (2019)

32. Seufert, M., Schatz, R., Wehner, N., Gardlo, B., Casas, P.: Is QUIC becoming the New TCP? On the Potential Impact of a New Protocol on Networked Multimedia QoE. In: 2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX) (2019)
33. Spang, B., Kunamalla, S., Teixeira, R., Huang, T.Y., Armitage, G., Johari, R., McKeown, N.: Sammy: Smoothing video traffic to be a friendly internet neighbor. p. 754–768. ACM SIGCOMM'23 (2023)
34. Thibaud, A., Fasson, J., Arnal, F., Pradas, D., Dubois, E., Chaput, E.: QoE enhancements on satellite networks through the use of caches. International Journal of Satellite Communications and Networking **36**, 553–565 (2018)
35. Thomas, L., Dubois, E., Kuhn, N., Lochin, E.: Google QUIC performance over a public SATCOM access. International Journal of Satellite Communications and Networking **37**, 601–611 (2019)
36. Wamser, F., Seufert, M., Casas, P., Irmer, R., Tran-Gia, P., Schatz, R.: YoMoApp: A tool for analyzing QoE of YouTube HTTP adaptive streaming in mobile networks. In: EuCNC'15. pp. 239–243 (2015)
37. Xu, S., Wang, X., Huang, M.: Modular and deep QoE/QoS mapping for multimedia services over satellite networks. International Journal of Communication Systems **31** (2018)
38. Yin, X., Jindal, A., Sekar, V., Sinopoli, B.: A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP. SIGCOMM Comput. Commun. Rev. p. 325–338 (2015)