THE WEB CONFERENCE ACM

Research Track | Systems and Infrastructure for Web, Mobile, and WoT

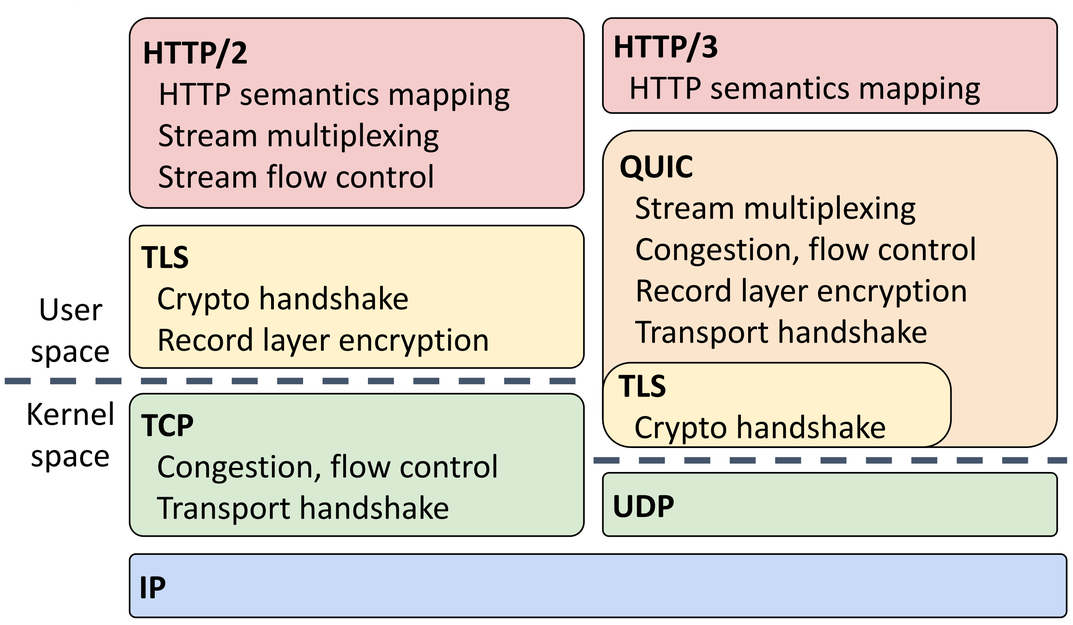
QUIC is not Quick Enough over Fast Internet

Xumiao Zhang[1], Shuowei Jin[1], Yi He[1], Ahmad Hassan[2], Z. Morley Mao[1], Feng Qian[2], Zhi-Li Zhang[3]

[1]University of Michigan [2]University of Southern California [3]University of Minnesota

1. Introduction and Background

QUIC is a user-space transport protocol over UDP. It is expected to be a game-changer in improving web application performance. Initially proposed and developed by Google (**gQUIC**) to enable fast, reliable, and secure connections, it was later refined by an IETF working group. They separate its fused transport, cryptographic handshakes, and HTTP functionalities into parts, leading to the standardization of **IETF QUIC**. As the application layer wrapper for QUIC, HTTP/3 was structured to integrate seamlessly with QUIC while maintaining compatibility with existing HTTP/2 functionalities. Together with the network layer and layers below, **UDP**, **QUIC**, and **HTTP/3** form a new protocol stack for the next-generation network communication, whose current counterpart is the stack of TCP, TLS, and HTTP/2.



QUIC's Benefits:
- 0/1-RTT Connection Establishment/Resumption
- Stream Multiplexing without Head-of-Line Blocking
- Integrated Security with TLS 1.3
- Connection Migration

2. Motivation and Challenges

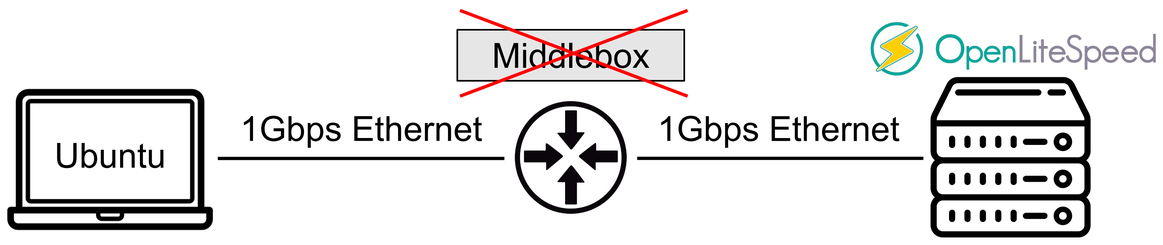
QUIC has attracted wide research attention. However, existing studies use diverse QUIC implementations (customized vs. commercial), compute environments (mobile vs. desktop), and network conditions (wired vs. wireless). Due to such diversity, their findings are a mixture of performance gains and degradations, compared to TCP or earlier generations of HTTP. Moreover, many of these studies focus on low-throughput use cases.

We advocate examining QUIC in the "**context**" of specific services and scenarios. In this study, we systematically study QUIC in an under-explored scenario, **running QUIC over high-speed networks**.

3. Methodology

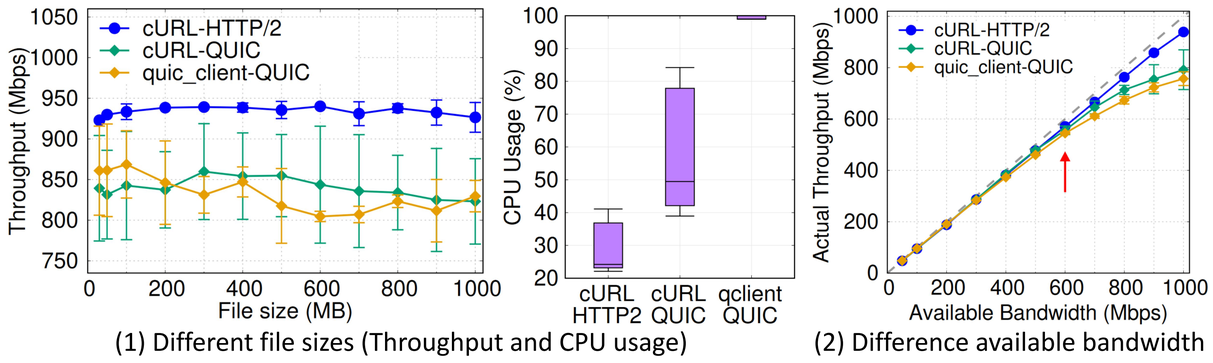
We conduct a series of experiments to compare the UDP+QUIC+HTTP/3 (**QUIC**) stack with the TCP+TLS+HTTP/2 (**HTTP/2**) stack. We focus on their performance in HTTP file download, video streaming, and web page loading.

Experimental Setup:
- Ubuntu 18.04 client and server
- 1 Gbps Ethernet, 2 hops away
- OpenLiteSpeed (v1.7.15, based on LSQUIC)
- Increased TCP/UDP buffer sizes to exceed link's BDP



4. Performance Comparison

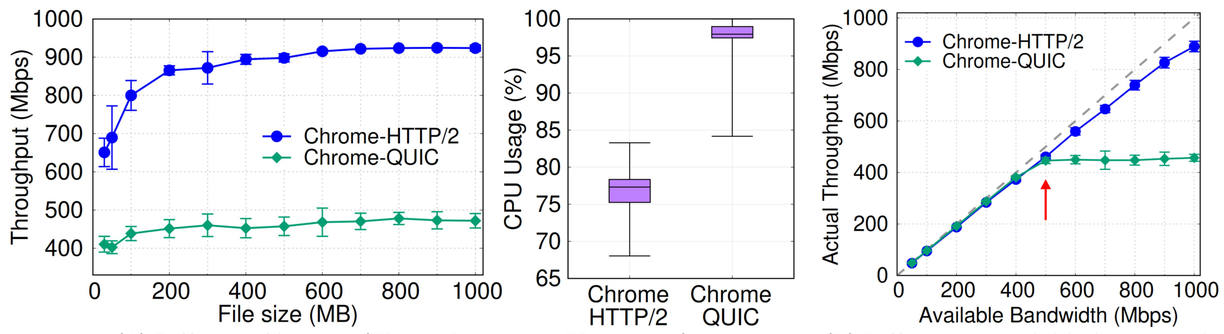
Exp. 1: File Download on Lightweight Clients (`cURL` and `quic_client`)
- Both tools running QUIC have lower throughput than cURL with HTTP/2.
- QUIC's CPU usage is also higher.
- QUIC loses to HTTP/2 at high throughput due to limited computation.



(1) Different file sizes (Throughput and CPU usage) (2) Difference available bandwidth

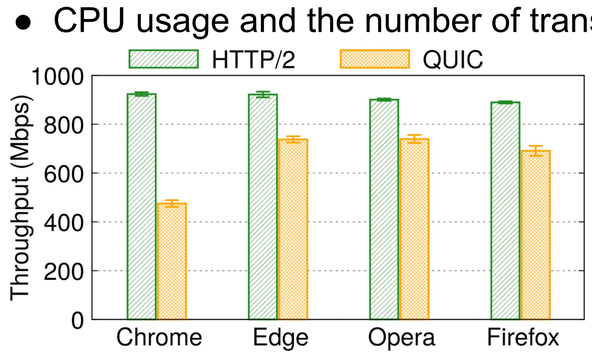
4. Performance Comparison (Cont.)

Exp. 2.1: File Download on Real Browsers (Chrome)
- A more significant performance gap between the two protocol stacks.
- QUIC fails to fully utilize the bandwidth starting earlier.



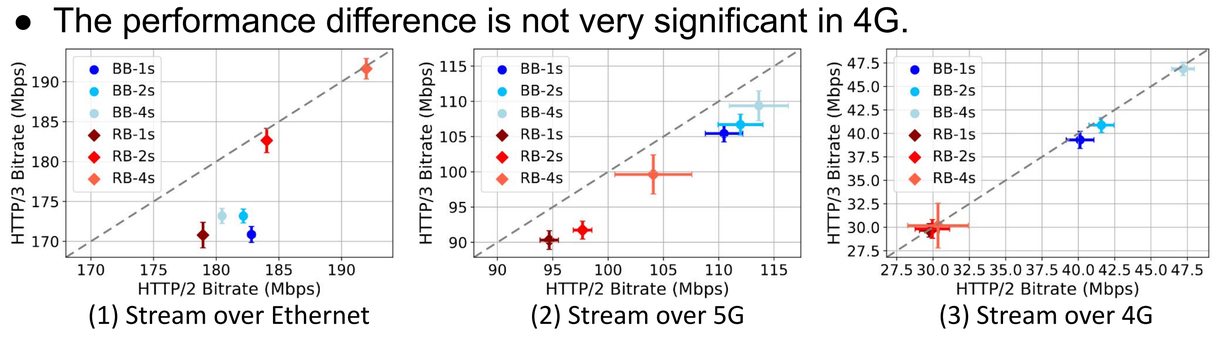
(1) Different file sizes (Throughput and CPU usage) (2) Difference available bandwidth

Exp. 2.2: File Download on Different QUIC-capable Browsers
Extend the HTTP file download experiments and Yield similar observations.
- All browsers have a degraded QUIC performance when QUIC is enabled.
- CPU usage and the number of transmitted packets increase for QUIC.



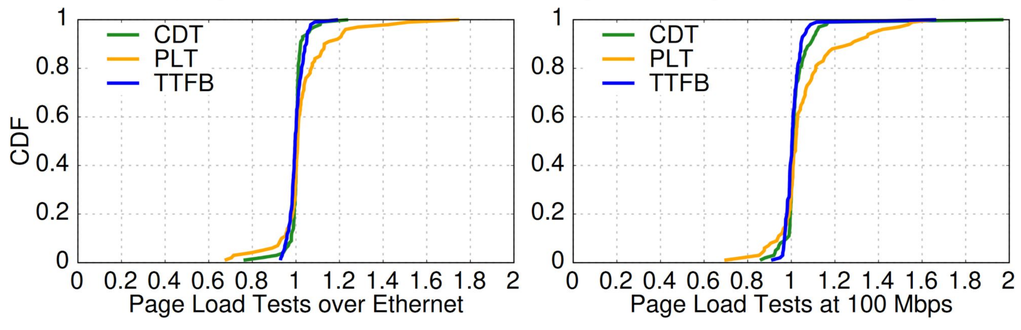
Browser	HTTP/2 CPU (%)	HTTP/3 CPU (%)
Chrome	77.1±4.7	97.4±4.5
Edge	28.4±3.9	81.1±7.7
Opera	27.9±3.3	84.9±14.5
Firefox	185.8±61.9	213.0±46.5

Exp. 3.1: Application Study - Video Streaming
4K video, 6 tracks, 3 chunk durations; 2 ABR algorithms (buffer/rate-based).
- QUIC performs worse than HTTP/2 in Ethernet and 5G scenarios.
- The performance difference is not very significant in 4G.



(1) Stream over Ethernet (2) Stream over 5G (3) Stream over 4G

Exp. 3.2: Application Study - Web Browsing
Web page load involves the transfer of multiple small objects.
- Most websites have not enabled QUIC yet.
- The performance gap is not as pronounced as that in the video tests.
- QUIC's page load time is 3.0% longer; Long tail of time gaps over 50%.



CDT: content download time

PLT: page load time

TTFB: time to first byte

5. Root Cause Analysis

We identify root causes for the QUIC-HTTP/2 performance gap.

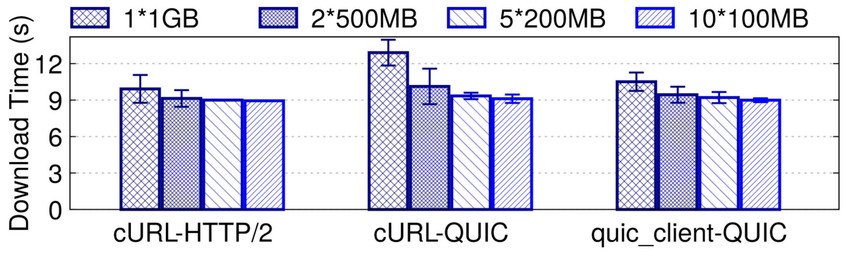
Eliminating Non-contributing Factors: server software, UDP/TCP, HTTP syntax, TLS encryption, parameter tuning, client OS, disk and memory.

Evidence from Packet Trace Analyses:
- QUIC perceives much more packets than HTTP/2 during file download.
- QUIC has much higher RTT dominated by local processing, 16.2 vs 1.9ms.

Root Cause Analysis via OS/Chromium Profiling:
- Fine-grained profiling in the kernel and user spaces reveals the "culprit".
- **QUIC's slowness is due to receiver-side processing**
  - Root causes: excessive data packets, QUIC's user-space ACKs

6. Recommendations for Mitigation

We make several recommendations for mitigating the observed issues.
- Adoption of UDP GRO (generic receive offload) on the Receiver Side
- QUIC-friendly Improvements to the offloading solution
- Optimizing QUIC logic on the Receiver Side
- Multi-threaded download for large files



Exp.: Parallel download

Increasing the number of instances reduces the download time.