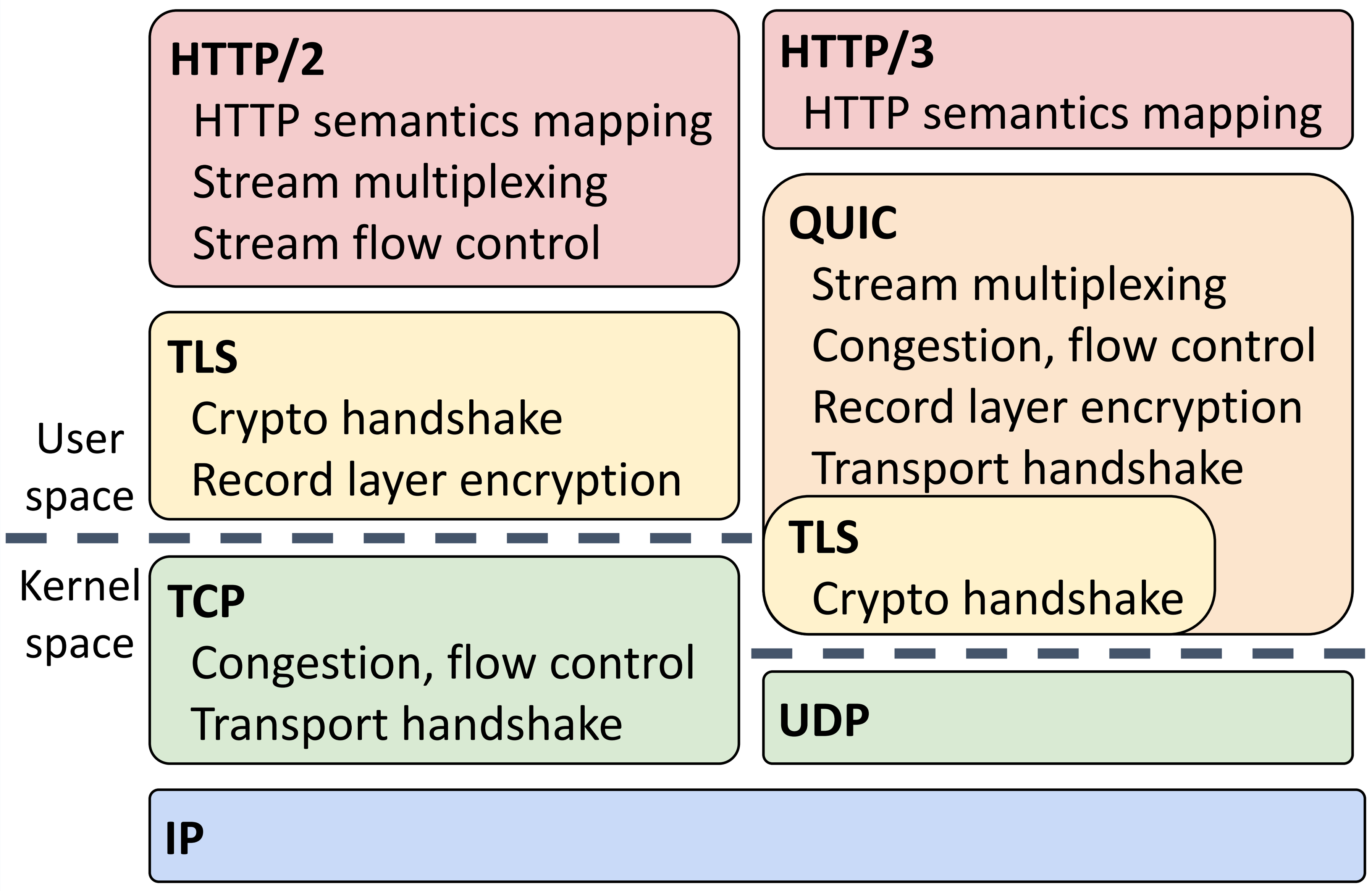
# QUIC is not Quick Enough over Fast Internet

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. 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 (TLS 1.3)
- Connection migration

## 2. Motivation and Challenges

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

We advocate examining QUIC in "**context**". We should also target a specific scenario, in this study, **running QUIC over high-speed networks**.

This scenario is becoming increasingly important:

- Emergence of high-speed networking (WiFi 6/7, 5G)
- Increasing deployment of QUIC (Google, Meta, …)
- Bandwidth-intensive applications (4K video, VR/AR)

Specifically, we aim to answer the following questions:

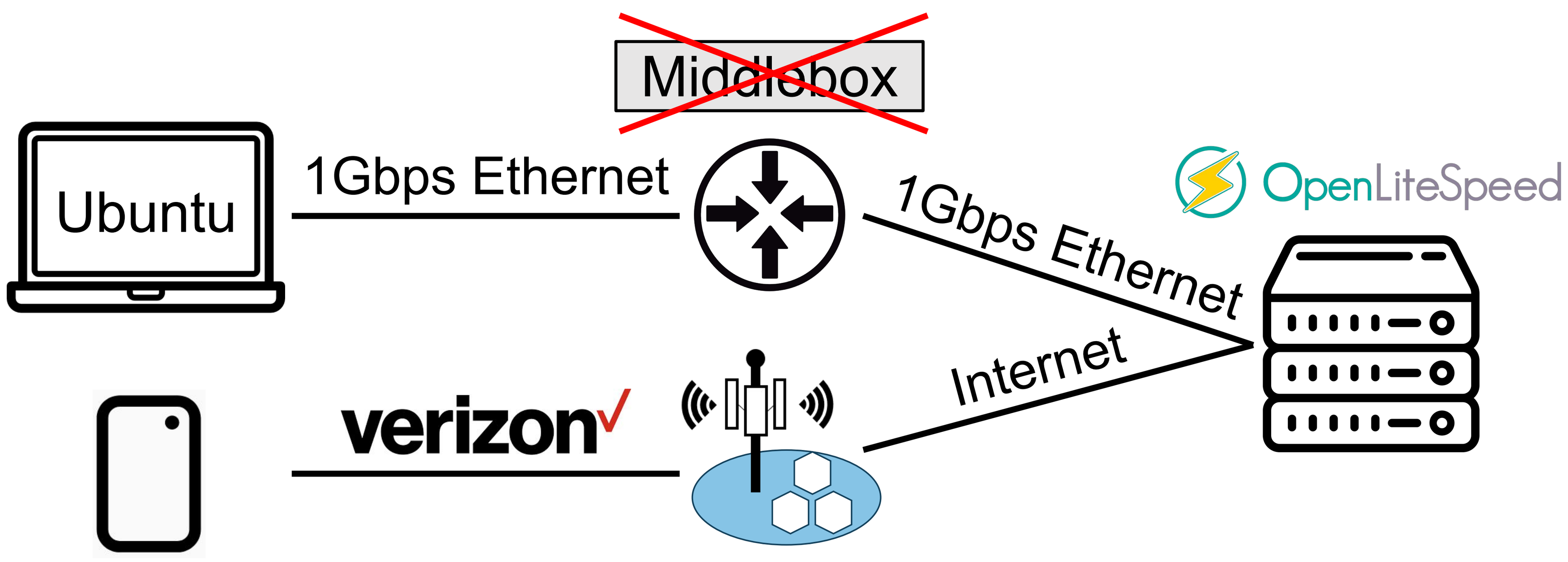
- When is QUIC data transfer slower than HTTP/2?
- What are the reasons for such a performance gap?
- Can we benefit from current deployment of QUIC?

## 3. Preliminary Results

We propose to **examine QUIC's performance over fast Internet**. We perform a series of experiments to compare the UDP+QUIC+HTTP/3 (**QUIC**) stack with the TCP+TLS+HTTP/2 (**HTTP/2**) stack.

**Experimental Setup**:

- Ubuntu 18.04 client and server; Pixel 5 phone
- 1Gbps Ethernet; low-band/mmWave 5G networks
- OpenLiteSpeed (v1.7.15) based on LSQUIC
- Increased buffer sizes to exceed link's BDP

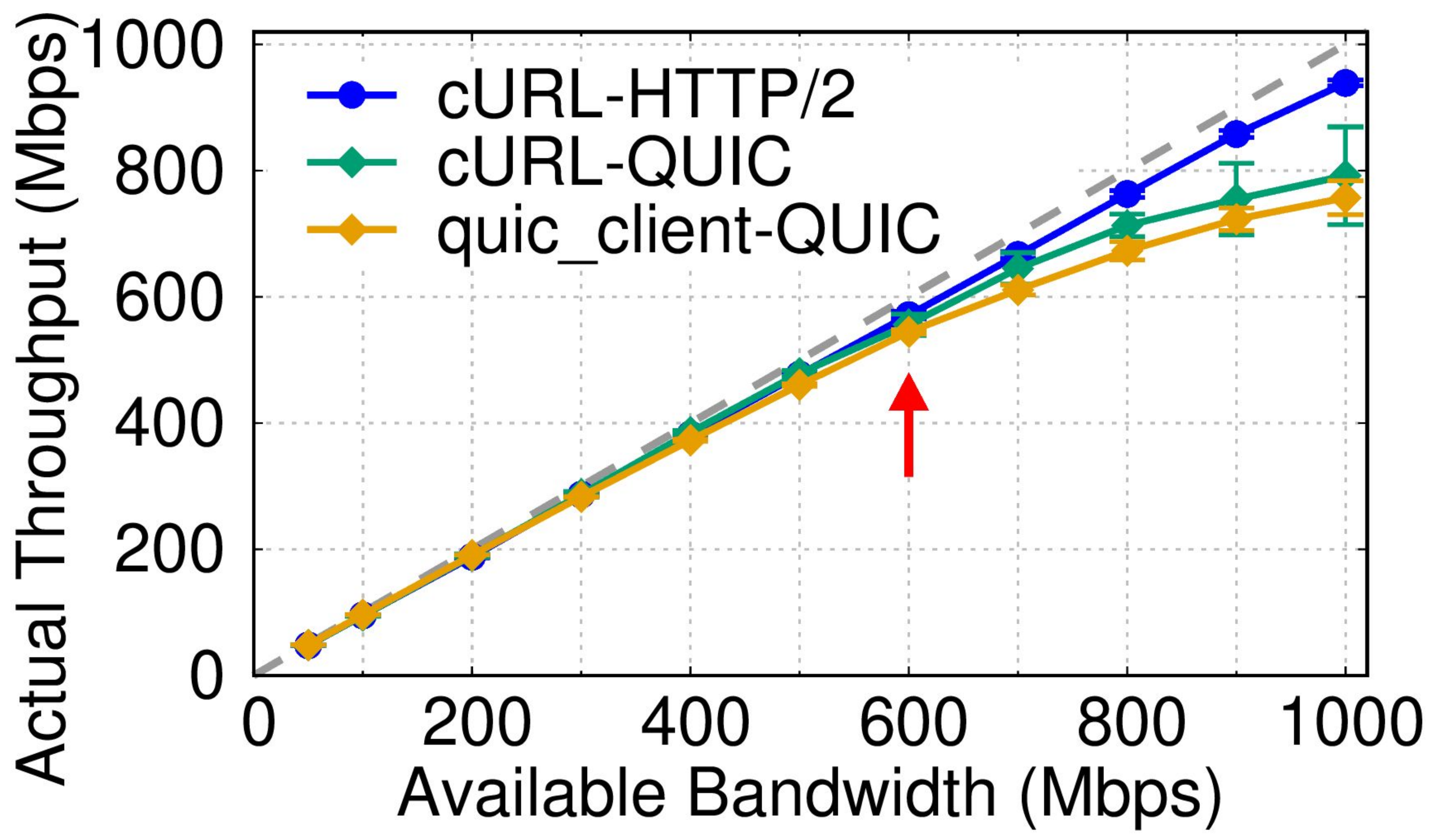



**Exp. 1**: 1GB file download using the Chrome browser. QUIC is slower than HTTP/2. It is worse on the phone.

| Testbed | Download Time (s) | | CPU Usage (%) | |
|---|---|---|---|---|
| | HTTP/2 | QUIC | HTTP/2 | QUIC |
| Desktop, Ethernet | 9.32 | 18.60 | 77.50 | 96.90 |
| Pixel 5, low-band 5G | 37.11 | 78.65 | 121.55 | 161.77 |
| Pixel 5, mmWave 5G | 30.10 | 63.20 | 128.43 | 165.20 |

CPU: Desktop - browser's network service; Phone - the entire browser process.

**Exp. 2**: 1GB file download in a simplified environment, using `cURL` and `quic_client` on the desktop, with changing bandwidth. When bandwidth is high (>600 Mbps), QUIC falls behind HTTP/2, by up to 15.7%.




## 4. Next Steps

We aim to comprehensively understand QUIC over fast Internet and identify root causes for its slowness.

- Experiments: different workloads, network types.
- Root cause analysis: application/OS profiling.
- Recommendations for mitigation