A Performance Perspective on Web Optimized Protocol Stacks: TCP+TLS+HTTP/2 vs. QUIC

Konrad Wolsing, Jan Rüth, Klaus Wehrle, Oliver Hohlfeld
Evolution of Web-stacks

- Multiplexing
- Encryption
- Congestion Control
- Loss Recovery

- Multiplexing
- Encryption
- Congestion Control
- Loss Recovery

- **HTTP/2**
- **TLS**
- **TCP**
- **IP**
- **gQUIC**
- **UDP**
- **H2 over gQUIC**

- **Evolvability over time (no ossification)**
- **No head-of-line blocking**
- **0-RTT connection establishment**

Konrad Wolsing
Related Work (QUIC vs. TCP)

**QUIC: Better For What And For Whom?**
Sarah Cook, Bernard Mathews, Patrick Troweg and Isabelle Ichimaru
Orange Labs
Lannion, France
Email: {sarah.cook, bernard.mathews, patrick.troweg, isabelle.ichimaru}@orange.com

**How quick is QUIC?**
Peter Maghsoodi, Zohdi Krämer, Sandor Molnar
High Speed Networks Lab, Dept. of Telecom., and Media Informatics, Budapest University of Technology and Economics, Budapest, Hungary
E-mail: magnsoodi, krazer, molnar@fres.univ.hu

**Taking a Long Look at QUIC**
An Approach for Rigorous Evaluation of Rapidly Evolving Transport Protocols
Arash Mirdadi Khosht Porud University
Samuel Jero
Northeastern University
cjero@purdue.edu
David Choffnes
Northeastern University
choffnes@cs.neu.edu

**QUIC and TCP: A Performance Evaluation**
Klára Németh
Centre de Informatics
Univ. of California, Berkeley, CA
kleins@alpaca.net
Igor Mirandia da Silva
Centre de Informatics
Univ. of California, Berkeley, CA
mirsil@berkeley.edu
Rafael Rafequ Ischoff
Dept. of Computer Science
Univ. of California, Berkeley, CA
rafequ@alpaca.net
Daniel Boix
Centre de Informatics
Univ. of California, Berkeley, CA
dboix@alpaca.net
Maria Sávia Jo
Centre de Informatics
Univ. of California, Berkeley, CA
msaiv@alpaca.net
Wesley Meiro
Centre de Informatics
Univ. of California, Berkeley, CA
wmeiro@alpaca.net
Djeziel Gadelha
Centre de Informatics
Univ. of California, Berkeley, CA
djeziel@alpaca.net

**Does QUIC make the Web faster?**
Praveen Biswal
Department of Computer Science
University of Houston
Praveen.Biswal@uh.edu

Omidraaf Ganswal
Department of Computer Science
University of Louisiana
Omidraaf.Ganswal@louisiana.edu

**when QUIC Meets TCP: an Experimental Study**
Yujin Yu,1 Mingswu Xu,1 Yun Yang1
1 Department of Computer Science and Technology, Tsinghua University, China
2 Tsinghua National Laboratory for Information Science and Technology
Email: yu.yujin@mails.tsinghua.edu.cn, xumwei@emrc.edu.cn, yunyang@cs.tsinghua.edu.cn

**Related Work (QUIC vs. TCP)**
Konrad Wolsing

---

**ABSTRACT**

This paper investigates “Quick UDP Internet Connections” (QUIC), which was proposed by Google in 2016 as a reliable protocol on top of UDP to reduce the latency incurred by the client-side TCP handshake and to efficiently transport data. QUIC, which is the latest protocol in a series developed by Google, is designed to enable a secure, efficient and scalable connection of web applications. In this paper, we compare QUIC with TCP, the widely used protocol for web traffic. We show that QUIC significantly reduces the latency and improves the throughput compared to TCP, especially in scenarios with low latency and high bandwidth.

**QUIC: Better For What And For Whom?**

Sarah Cook, Bernard Mathews, Patrick Troweg and Isabelle Ichimaru
Orange Labs
Lannion, France
Email: {sarah.cook, bernard.mathews, patrick.troweg, isabelle.ichimaru}@orange.com

**How quick is QUIC?**

Peter Maghsoodi, Zohdi Krämer, Sandor Molnar
High Speed Networks Lab, Dept. of Telecom., and Media Informatics, Budapest University of Technology and Economics, Budapest, Hungary
E-mail: magnsoodi, krazer, molnar@fres.univ.hu

**Taking a Long Look at QUIC**

Arash Mirdadi Khosht Porud University
Samuel Jero
Northeastern University
cjero@purdue.edu
David Choffnes
Northeastern University
choffnes@cs.neu.edu

**QUIC and TCP: A Performance Evaluation**

Klára Németh
Centre de Informatics
Univ. of California, Berkeley, CA
kleins@alpaca.net
Igor Mirandia da Silva
Centre de Informatics
Univ. of California, Berkeley, CA
mirsil@berkeley.edu
Rafael Rafequ Ischoff
Dept. of Computer Science
Univ. of California, Berkeley, CA
rafequ@alpaca.net
Daniel Boix
Centre de Informatics
Univ. of California, Berkeley, CA
dboix@alpaca.net
Maria Sávia Jo
Centre de Informatics
Univ. of California, Berkeley, CA
msaiv@alpaca.net
Wesley Meiro
Centre de Informatics
Univ. of California, Berkeley, CA
wmeiro@alpaca.net
Djeziel Gadelha
Centre de Informatics
Univ. of California, Berkeley, CA
djeziel@alpaca.net

**Does QUIC make the Web faster?**

Praveen Biswal
Department of Computer Science
University of Houston
Praveen.Biswal@uh.edu

Omidraaf Ganswal
Department of Computer Science
University of Louisiana
Omidraaf.Ganswal@louisiana.edu

**when QUIC Meets TCP: an Experimental Study**

Yujin Yu,1 Mingswu Xu,1 Yun Yang1
1 Department of Computer Science and Technology, Tsinghua University, China
2 Tsinghua National Laboratory for Information Science and Technology
Email: yu.yujin@mails.tsinghua.edu.cn, xumwei@emrc.edu.cn, yunyang@cs.tsinghua.edu.cn

---

**Related Work (QUIC vs. TCP)**
Konrad Wolsing
Related Work (QUIC vs. TCP)

- **Unoptimized TCP stacks**
  - QUIC is optimized for web performance
  - TCP can be tuned too
  - Related work does not tune TCP

- **Connection establishment**
  - QUIC requires 0–1 RTT
  - TCP+TLS usually 2 RTT

- **User-centered metrics**
  - research focuses on PLT
  - PLT is not suited for user perception

Our Goal
Reproducible, user-centered performance evaluation on eye-to-eye level between TCP+TLS+HTTP/2 and QUIC.
Achieving comparability

• **TCP + TLS + HTTP**
  - ✔ Initial window 32
  - ✔ Packet pacing
  - ★ 1-RTT handshake
  - ★ Use TLS1.3 without 0-RTT
  - ✔ Use only HTTP/2
  - ✗ Head-of-line blocking

• **QUIC**
  - ✔ Initial window 32
  - ✔ Packet pacing
  - ★ no 0-RTT connection setup
  - ✔ No encryption overhead
  - ✔ HTTP/2
  - ✔ No head-of-line blocking

• **Protocol settings**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>Stock TCP (Linux 4.18): IW10, Cubic</td>
</tr>
<tr>
<td>TCP+</td>
<td>IW32, Pacing, tuned network buffers, no slow start after idle, Cubic</td>
</tr>
<tr>
<td>TCP+BBR</td>
<td>TCP+, but with BBR as congestion control</td>
</tr>
<tr>
<td>QUIC</td>
<td>Google QUIC Version 43: IW32, Pacing, Cubic</td>
</tr>
<tr>
<td>QUIC+BBR</td>
<td>QUIC, but with BBR as congestion control</td>
</tr>
</tbody>
</table>
### Server speed test

- **QUIC 10MB**
- **NGINX 10MB**
- **QUIC 1MB**
- **NGINX 1MB**
- **QUIC 10KB**
- **NGINX 10KB**
- **QUIC 2B**
- **NGINX 2B**

### Network configuration

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Uplink</strong></td>
<td>5 Mbps</td>
<td>2.8 Mbps</td>
<td>3.54 Mbps</td>
<td>.468 Mbps</td>
<td>1.89 Mbps</td>
</tr>
<tr>
<td><strong>Downlink</strong></td>
<td>25 Mbps</td>
<td>10.5 Mbps</td>
<td>3.54 Mbps</td>
<td>.468 Mbps</td>
<td>1.89 Mbps</td>
</tr>
<tr>
<td><strong>RTT</strong></td>
<td>24 ms</td>
<td>74 ms</td>
<td>94 ms</td>
<td>262 ms</td>
<td>761 ms</td>
</tr>
<tr>
<td><strong>Loss</strong></td>
<td>0.0%</td>
<td>0.0%</td>
<td>.048%</td>
<td>3.3%</td>
<td>6.0%</td>
</tr>
<tr>
<td><strong>Queue size</strong></td>
<td>12 ms</td>
<td>200 ms</td>
<td>200 ms</td>
<td>200 ms</td>
<td>200 ms</td>
</tr>
</tbody>
</table>
Measurement

- **Replayed Websites**

  - 38 websites [8] from Alexa and Moz lists

  \[
  \text{performance gain}_{\text{QUIC}}^{\text{TCP}} = \frac{X_{\text{QUIC}} - X_{\text{TCP}}}{X_{\text{TCP}}}
  \]

  \[
  \frac{0.5s_{\text{QUIC}} - 0.75s_{\text{TCP}}}{0.75s_{\text{TCP}}} = -0.333
  \]

- **Visual Metrics**

  - Only PLT not above-the-fold

Konrad Wolsing
Performance gain DSL

- Tuning ⇒ increased performance
- QUIC outperforms even tuned TCP
- Steep curves: website size/structure has low impact
Performance gain LTE

Network LTE

• Results similar to DSL
• But differences are slightly larger
• Tuning impact reduced
• Variability rises
Performance gain DA2GC

- Tuning becomes a coin toss
- QUIC performs still good
  - No head-of-line blocking
  - Larger SACK ranges

Network DA2GC

(CDF)

(Downlink 468 Mbps, Uplink 468 Mbps, RTT 262 ms, Loss 3.3%, [7])
• Congestion control impacts performance
• BBR outperforms CUBIC
• QUIC with CUBIC still faster than TCP with CUBIC
Mean Performance Gain

- Mean performance gain
- TCP, TCP+, QUIC, TCP+BBR, QUIC, QUIC+BBR
- FVC, VC85, PLT
- DSL, LTE, 3G, MSS, DA2GC

Konrad Wolsing
Influence of Resources

![Graph showing the influence of various resources on network performance metrics such as 3G (QUIC - TCP+), DA2GC (QUIC+BBR - TCP+BBR), and MSS (QUIC+BBR - TCP+BBR). The graph compares different metrics like difference [s], FVC, SI, and VC85.](image-url)
### Subtracting Design Differences

Mean difference under PLT with one subtracted RTT

\[
\text{QUIC} - (\text{TCP} + 1 \cdot \text{RTT})
\]

<table>
<thead>
<tr>
<th>Net</th>
<th>Website</th>
<th>[ms]</th>
<th>[RTT]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSL</td>
<td>gnu.org</td>
<td>1.6</td>
<td>0.066</td>
</tr>
<tr>
<td>DSL</td>
<td>wikipedia.org</td>
<td>-3.1</td>
<td>-0.128</td>
</tr>
<tr>
<td>LTE</td>
<td>gnu.org</td>
<td>-30</td>
<td>-0.412</td>
</tr>
<tr>
<td>LTE</td>
<td>wikipedia.org</td>
<td>-13</td>
<td>-0.175</td>
</tr>
<tr>
<td>3G</td>
<td>gnu.org</td>
<td>-32</td>
<td>-0.344</td>
</tr>
<tr>
<td>3G</td>
<td>wikipedia.org</td>
<td>-54</td>
<td>-0.570</td>
</tr>
<tr>
<td>DA2GC</td>
<td>gnu.org</td>
<td>39</td>
<td>0.150</td>
</tr>
<tr>
<td>DA2GC</td>
<td>wikipedia.org</td>
<td>-1005</td>
<td>-3.834</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Net</th>
<th>Website</th>
<th>[ms]</th>
<th>[RTT]</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS</td>
<td>gnu.org</td>
<td>-1100</td>
<td>-1.447</td>
</tr>
<tr>
<td>MSS</td>
<td>wikipedia.org</td>
<td>-529</td>
<td>-0.696</td>
</tr>
</tbody>
</table>

**Congestion Control: BBR**

<table>
<thead>
<tr>
<th>Net</th>
<th>Website</th>
<th>[ms]</th>
<th>[RTT]</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS</td>
<td>gnu.org</td>
<td>-477</td>
<td>-0.628</td>
</tr>
<tr>
<td>MSS</td>
<td>wikipedia.org</td>
<td>451</td>
<td>0.593</td>
</tr>
</tbody>
</table>
Conclusion

• Results
  ▶ TCP tuning is not negligible
  ▶ Still QUIC outperforms TCP, but the gap gets narrower
  ▶ QUIC mostly faster due to the RTT reduced connection establishment
  ▶ Congestion control sometimes matters more than protocol choice

• Discussion
  ▶ QUIC is not build to primarily improve performance
  ▶ QUIC enables an evolvable stack especially on transport layer
  ▶ Open question: Do users perceive QUIC as faster?

Thank you for your attention.
P. Biswal and O. Gnawali.  
Does quic make the web faster?  
In *Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2016.

Breitbandmessung.  
Breitbandmessung Ergebnisse als interaktive Darstellung.  

G. Carlucci, L. De Cicco, and S. Mascolo.  
HTTP over UDP: an Experimental Investigation of QUIC.  

S. Cook, B. Mathieu, P. Truong, and I. Hamchaoui.  
QUIC: Better for what and for whom?  

How quick is QUIC?

Mile High WiFi: A First Look At In-Flight Internet Connectivity.

HTTP/2 Prioritization and its Impact on Web Performance.

When QUIC meets TCP: An experimental study.

Y. Zheng, Y. Wang, M. Rui, A. Palade, S. Sheehan, and E. O'Nuallain.
Performance Evaluation of HTTP/2 over TLS+ TCP and HTTP/2 over QUIC in a Mobile Network.