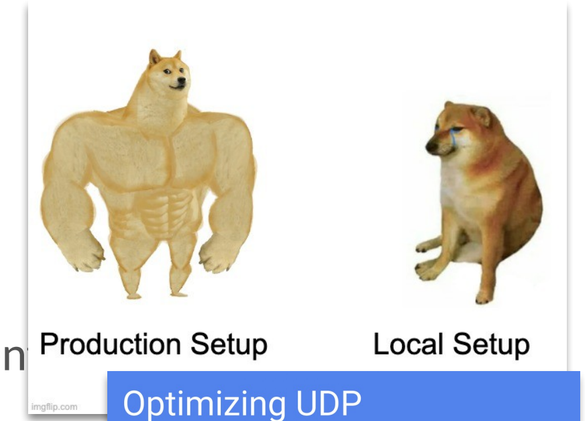
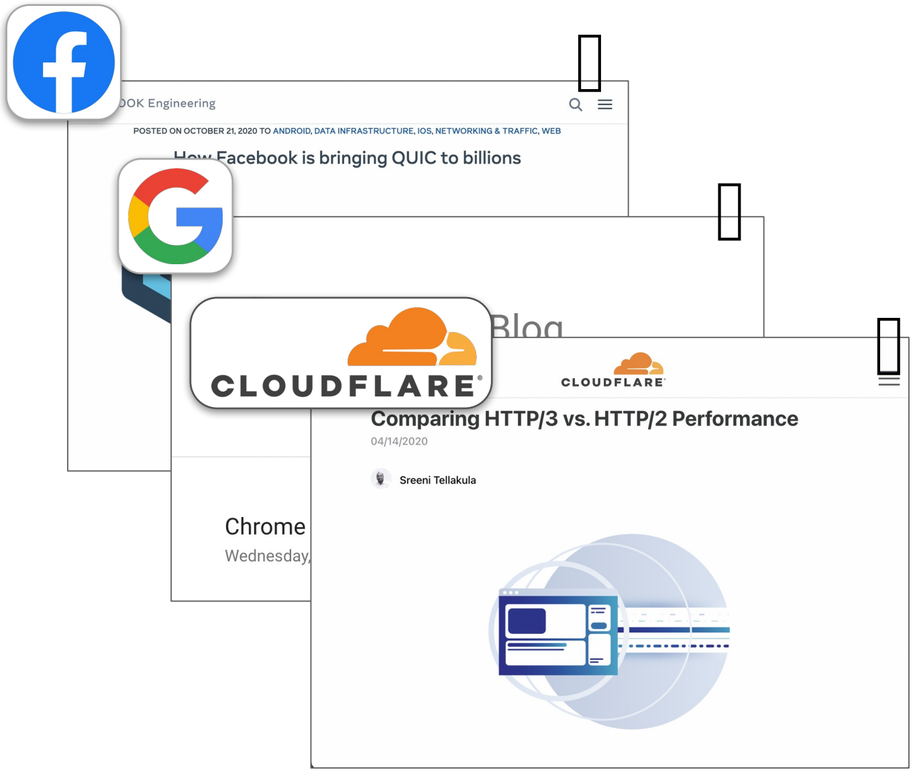
Dissecting Performance of Production QUIC

Theophilus A. Benson
Carnegie Mellon University

Since QUIC standardization began…



Various performance studies have been published by content providers and independen



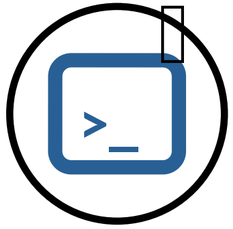
Optimizing UDP
for content delivery:
GSO, pacing and zerocopy

Author	Result	Ex
Saif et al [1]	In a local environment, QUIC performed **worse** than TCP for all network conditions except high loss.	Coc unoptimized open-source implementation.
Codavel [2]	During packet loss, QUIC performed **better** for small payloads (250KB) but **substantially worse** for large payloads (17MB).	Unoptimized open-source implementation and different congestion control.

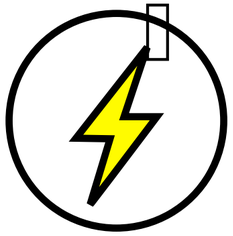
[1] Darius Saif, et al. 2020. An Early Benchmark of Quality of Experience Between HTTP/2 and HTTP/3 using Lighthouse.
[2] https://blog.codavel.com/how-fast-is-quic-and-what-makes-bolina-faster-pt-ii

Content providers' results are mixed and they do not offer much detailed root cause analysis.

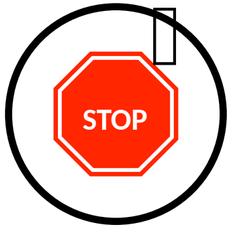
Researchers' results do not align with that of content providers.



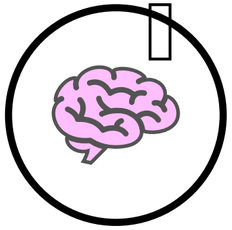
User-space Implementation



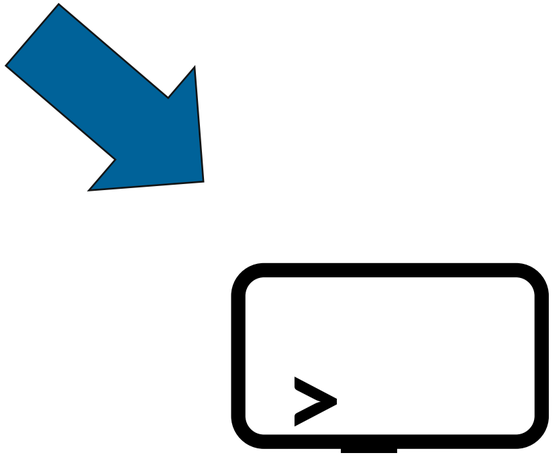
Faster Connection Establishment



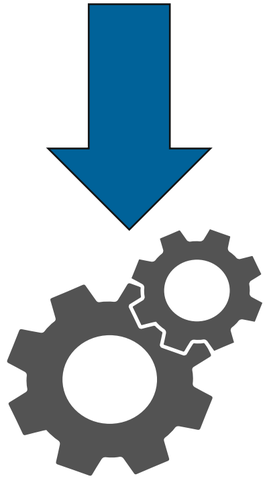
Removal of TCP's Head-of-Line (HOL) Blocking Problem



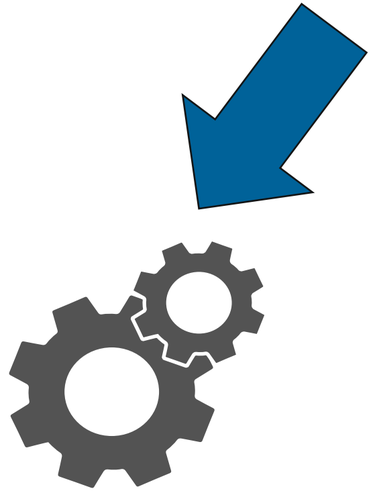
Improved and Simplified Recovery Mechanisms



Code

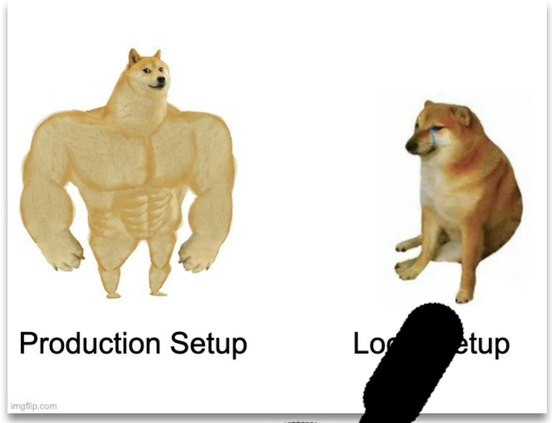


Optimization



Config.

[1] https://blog.chromium.org/2013/06/experimenting-with-quic.html



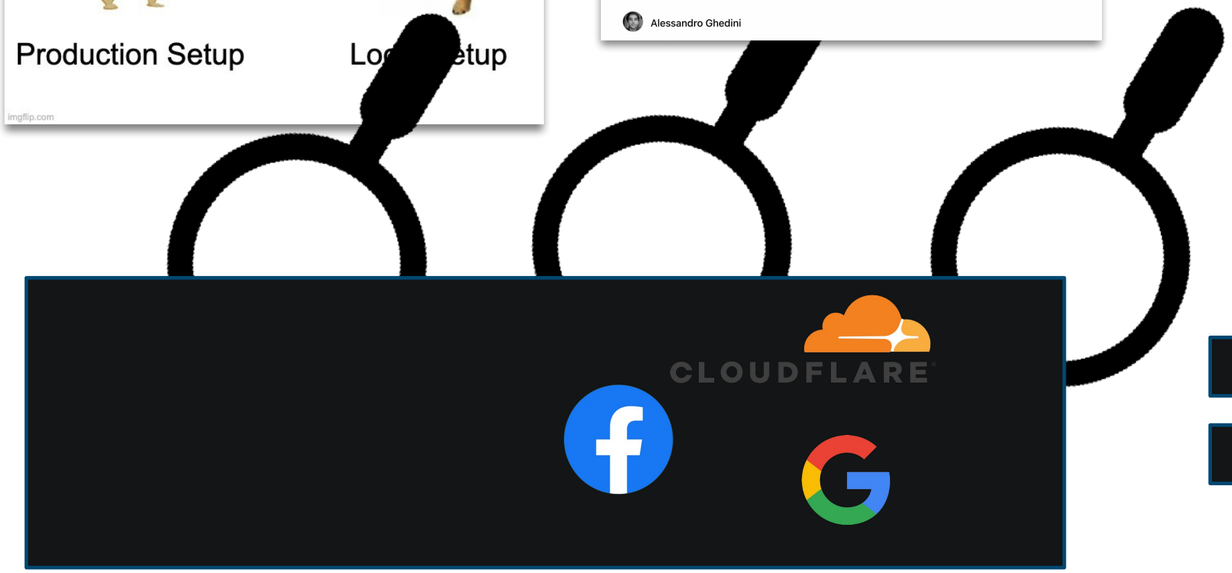
Optimizing UDP
for content delivery:
GSO, pacing and zerocopy

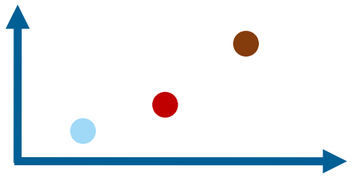
CLOUDFLARE

Accelerating UDP packet transmission for QUIC

01/08/2020

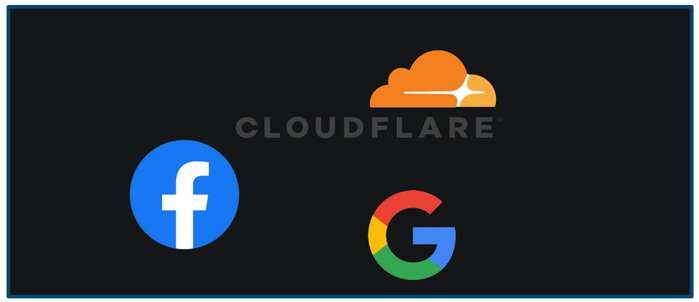
Alessandro Ghedini





Optimization Config. Code

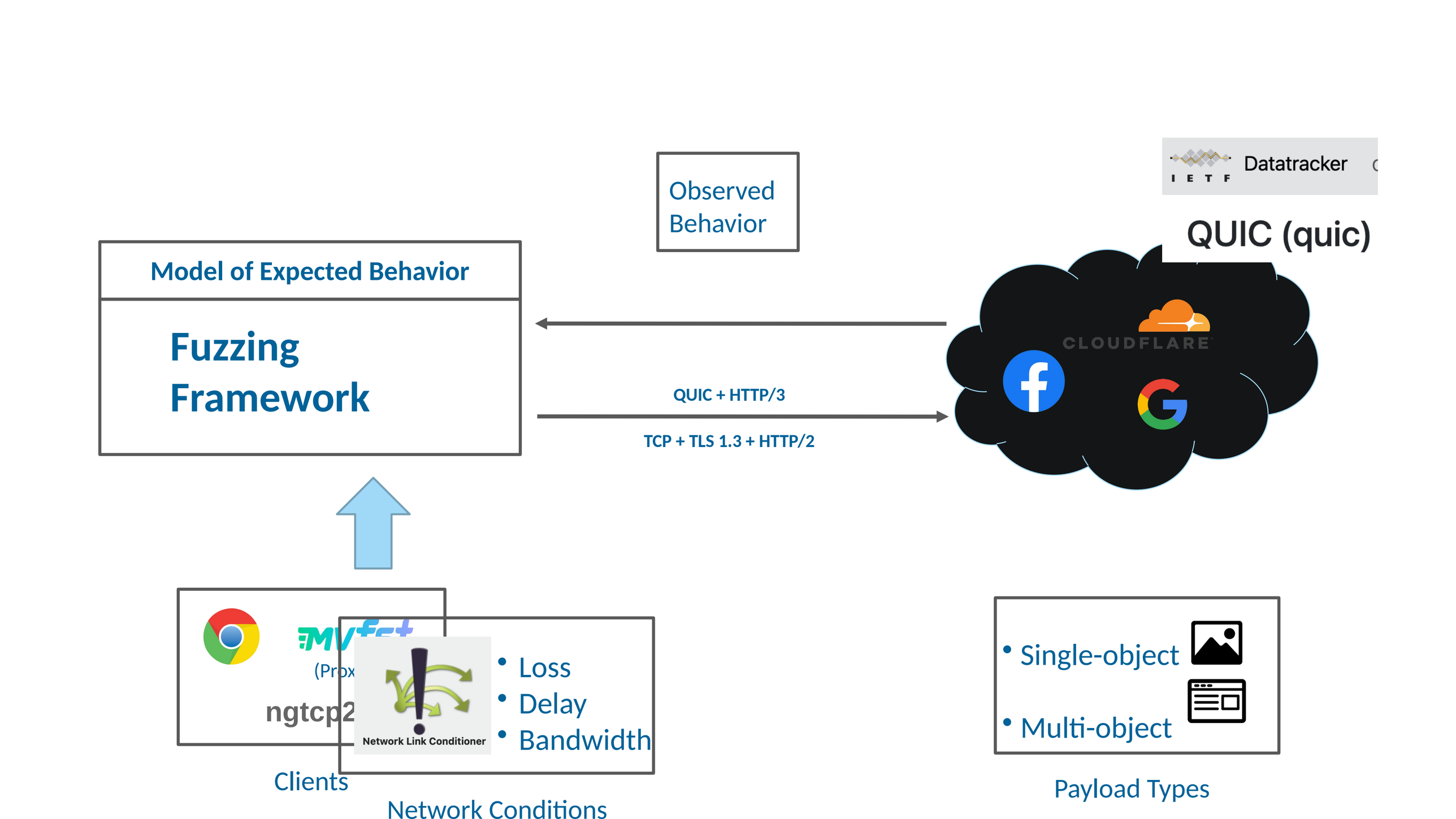
Performance Diagnosis Queries
To detect problems



Configuration Code

Optimization

New domain-specific models
To compensate for missing data


Datatracker
IETF
QUIC (quic)
Observed Behavior
Model of Expected Behavior
Fuzzing Framework
QUIC + HTTP/3
TCP + TLS 1.3 + HTTP/2
CLOUDFLARE
(Prox
ngtcp2
Network Link Conditioner
• Loss
• Delay
• Bandwidth
Clients
Network Conditions
• Single-object
• Multi-object
Payload Types

Insights from our study:

Faster Connection Establishment

Limited but noticeable impact

TLS implementation on client and server

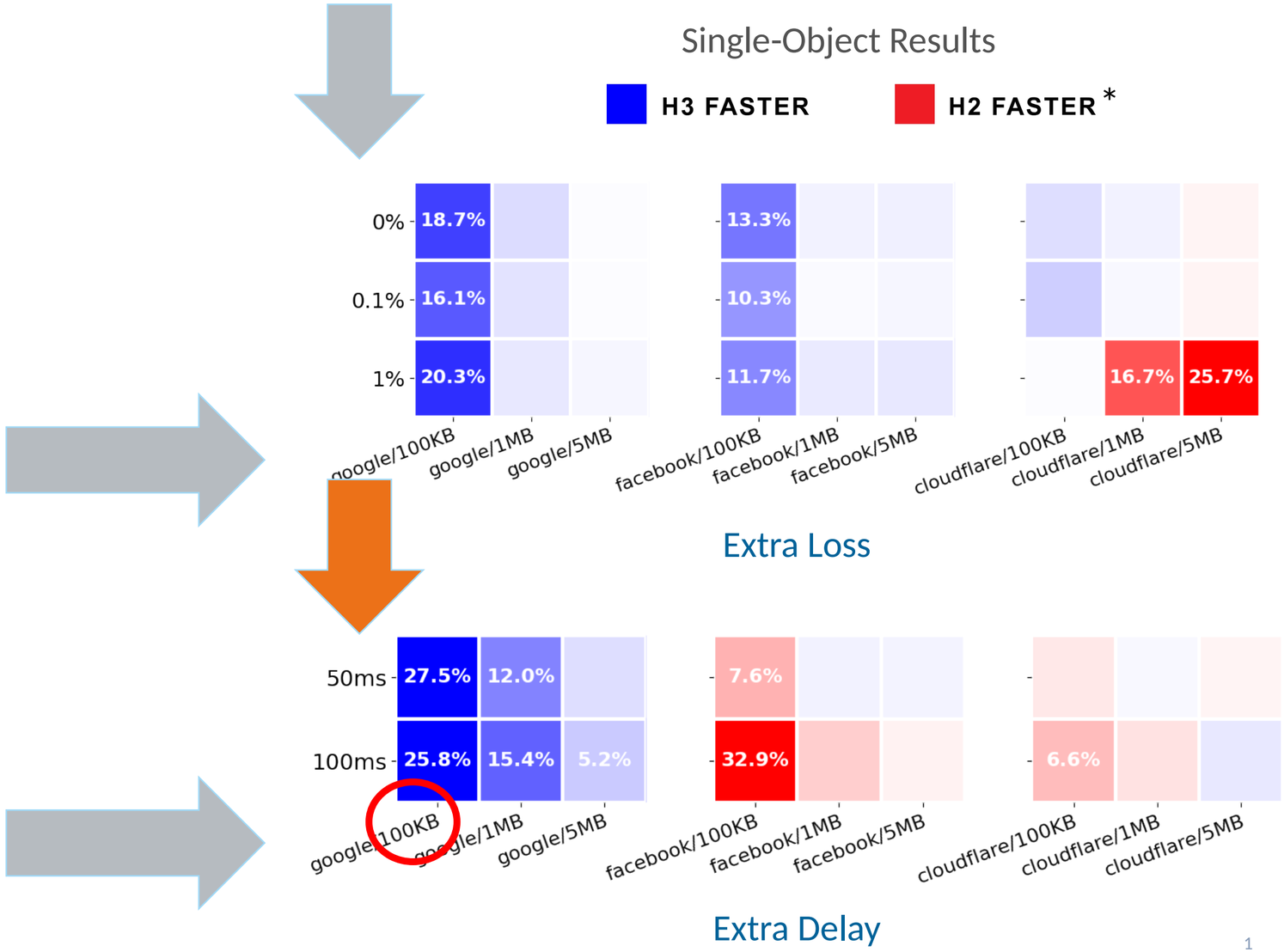
Coordination required for faster connection est.

Congestion Control (re-)Implementation

Qualify is important <u>but varies</u> significantly across providers

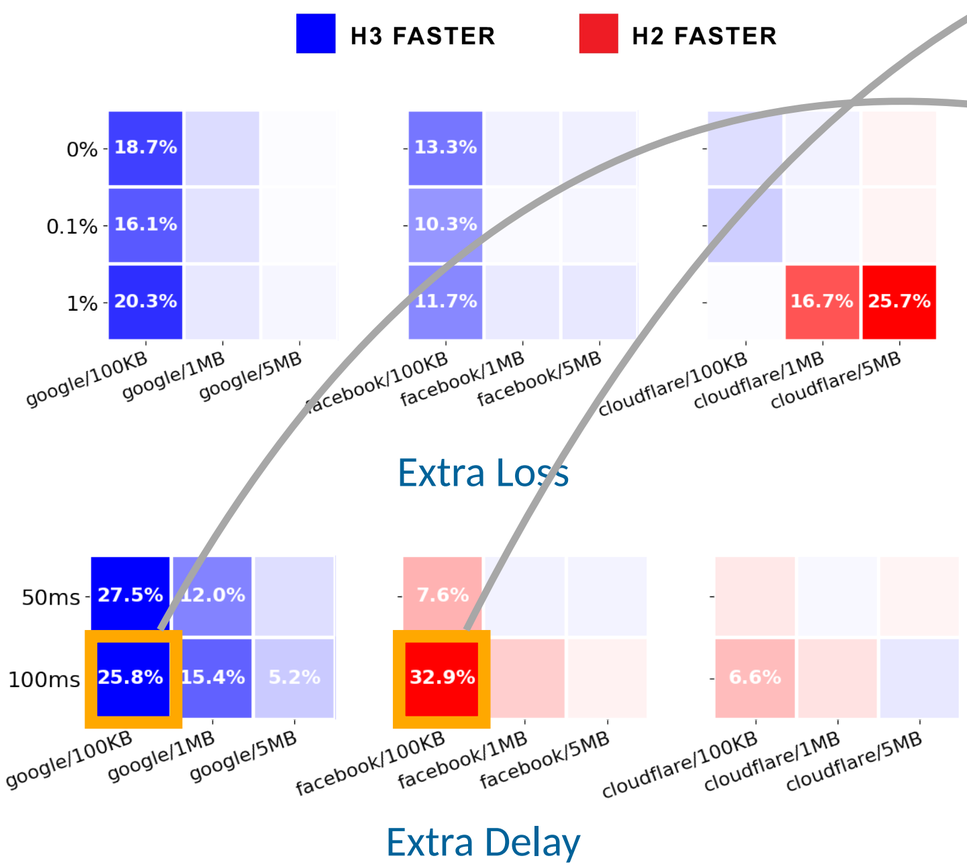
Removal of head of line blocking

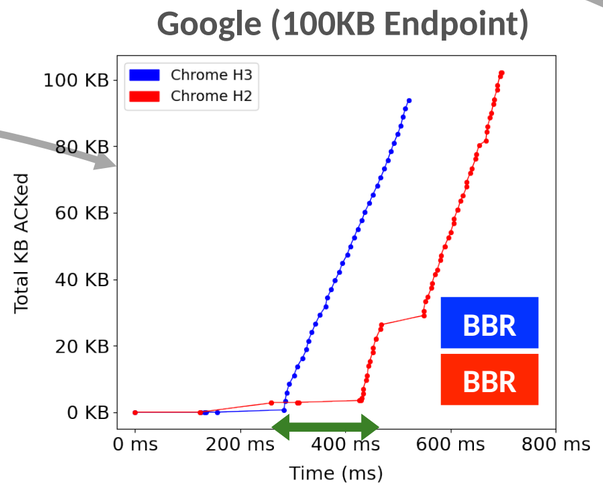
Unclear advances in practice

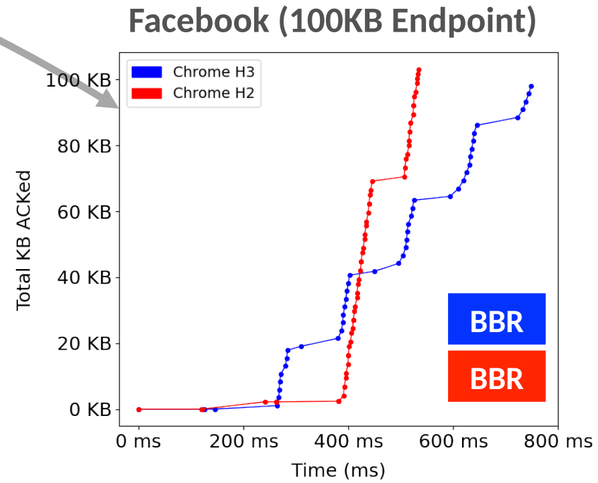


*Percentages < 5% are not labelled

Insight #2: The main reason why QUIC implementations differ in performance is because of congestion control heterogeneity (cont.)





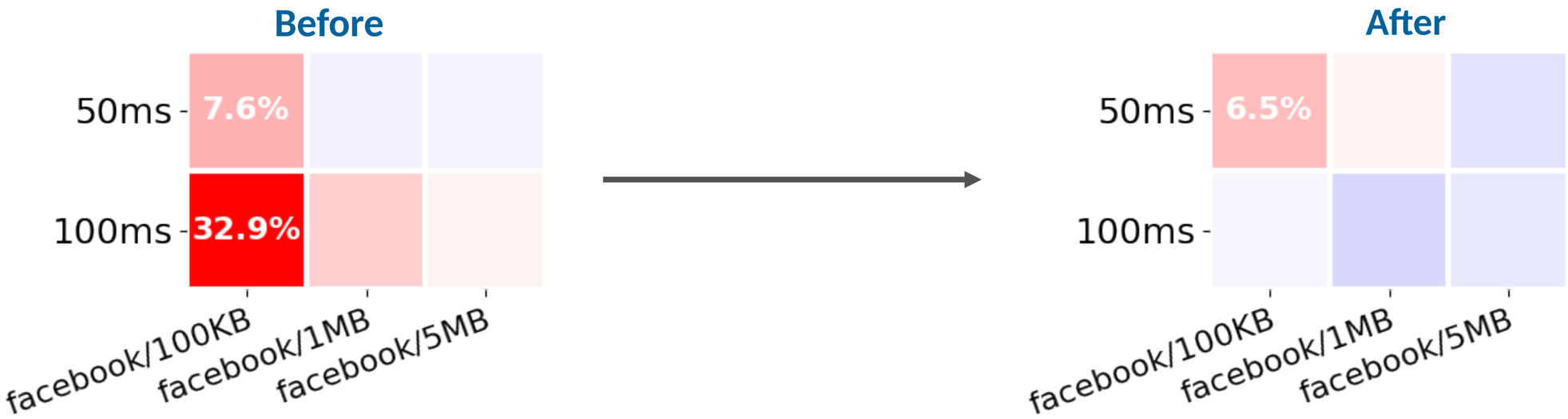


Bytes ACKed over time for Google and Facebook 100KB endpoints during 100ms added delay

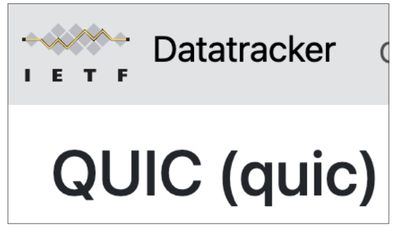
Key difference b/t Google and Facebook: **Bug** in Facebook's congestion control logic.

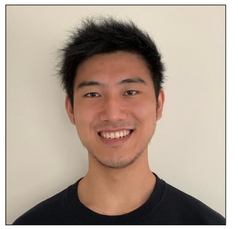
After we identified this bug, we notified the maintainers of Facebook Proxygen.

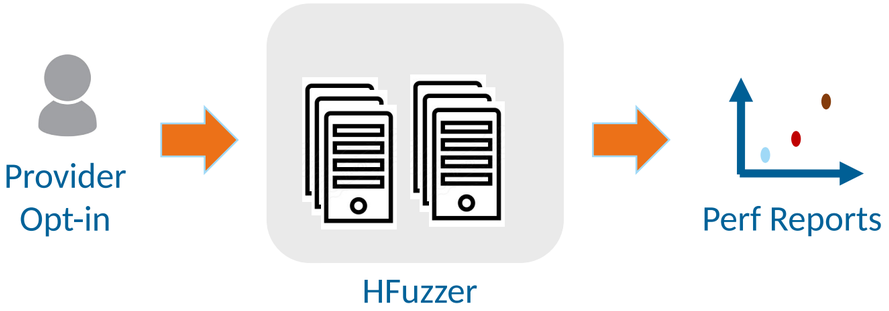
Solution: Simply ignore RTT measurements from “implicit” ACKs



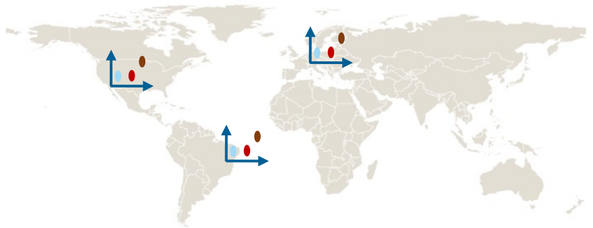
The Road Ahead .....







Automated Cloud Service



Region Specific Testing

Automated Diagnosis

theophilus@cmu.edu
@dottheophilus