

QUIC Steps: Evaluating Pacing Strategies in QUIC Implementations

Marcel Kempf¹, Simon Tietz¹, Benedikt Jaeger¹, Johannes Späth¹, Georg Carle¹, Johannes Zirngibl²

Friday 25th July, 2025

IETF 123 - IRTF maprg session
Madrid, Spain



¹ Technical University of Munich

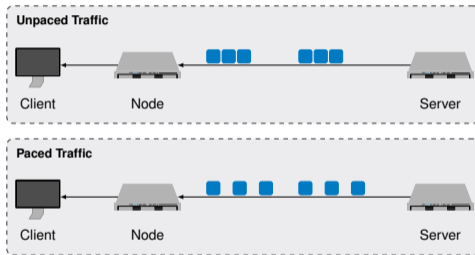


MAX PLANCK INSTITUTE
FOR INFORMATICS

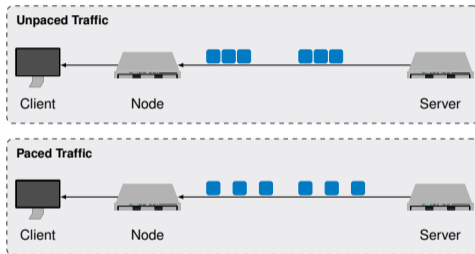
² Max Planck Institute for Informatics

- What is pacing?
- Do we really need pacing?
- And where's the problem now?

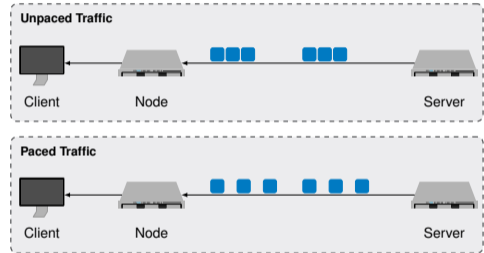
- What is pacing?
 - Action: Evenly distribute packets over time
 - Goals: Reduce burstiness, don't overwhelm middleboxes
 - Result: (Hopefully) less packet loss
- Do we really need pacing?
- And where's the problem now?



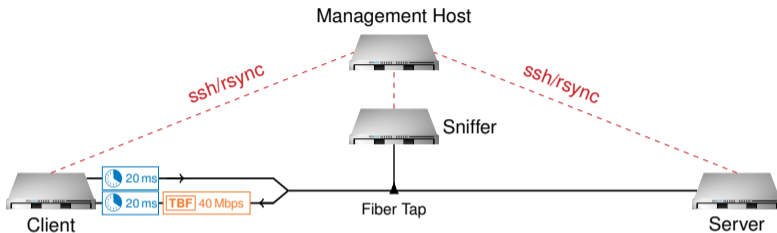
- What is pacing?
 - Action: Evenly distribute packets over time
 - Goals: Reduce burstiness, don't overwhelm middleboxes
 - Result: (Hopefully) less packet loss
- Do we really need pacing?
 - "A sender SHOULD pace sending of all in-flight packets [...]"
— RFC 9002, Section 7.7
 - Latency-sensitive apps (video calls, streaming) do not like bursts
 - Some congestion control algorithms (e.g., BBR) require pacing
- And where's the problem now?



- What is pacing?
 - Action: Evenly distribute packets over time
 - Goals: Reduce burstiness, don't overwhelm middleboxes
 - Result: (Hopefully) less packet loss
- Do we really need pacing?
 - “A sender SHOULD pace sending of all in-flight packets [...]”
— *RFC 9002, Section 7.7*
 - Latency-sensitive apps (video calls, streaming) do not like bursts
 - Some congestion control algorithms (e.g., BBR) require pacing
- And where's the problem now?
 - QUIC runs in user space
 - Coarse timers, syscall overhead, OS scheduling delays, ...
 - What if we use `sendmmsg()` or segmentation offloading (GSO)?



- How is pacing implemented in different QUIC libraries?
- How well do those strategies perform in practice?
- How can QUIC pacing benefit from kernel features?
- What happens when we use GSO and can we fix it?



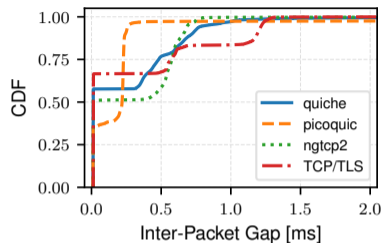
Library	Language	Developer	Send Time Calculation	Pacing Responsibility
quiche	Rust	Cloudflare	Last timestamp and pacing rate	Kernel (via SO_TXTIME)
picoquic	C	C. Huitema	Last timestamp and pacing rate	Application
ngtcp2	C	T. Tsujikawa	Credit-based approach (cf. RFC 9002)	Application

Evaluation – Baseline



Inter-Packet Gap (IPG)

- Distributions differ
- Roughly 50 % of the packets are sent back-to-back



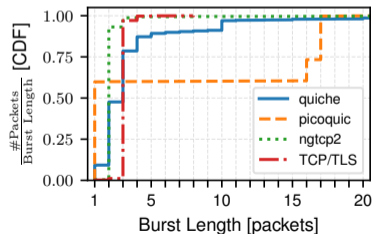
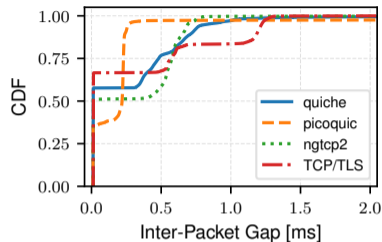
Evaluation – Baseline

Inter-Packet Gap (IPG)

- Distributions differ
- Roughly 50 % of the packets are sent back-to-back

Burst Length (BL)

- Burst = consecutive packets with an IPG of < 0.1 ms
- Strange behavior uncovered
 - quiche seems to be quite unstable
 - picoquic seems to send either single packets or large bursts



Evaluation – Baseline

Inter-Packet Gap (IPG)

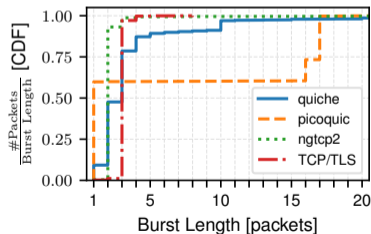
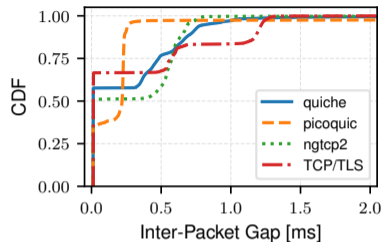
- Distributions differ
- Roughly 50 % of the packets are sent back-to-back

Burst Length (BL)

- Burst = consecutive packets with an IPG of < 0.1 ms
- Strange behavior uncovered
 - quiche seems to be quite unstable
 - picoquic seems to send either single packets or large bursts

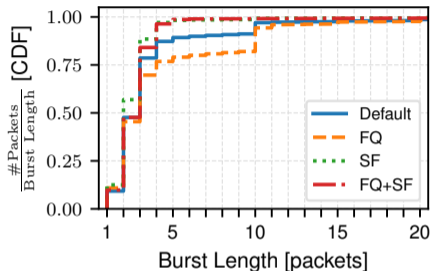
Take-away Result:

- Large differences in the distributions of IPGs and BLs

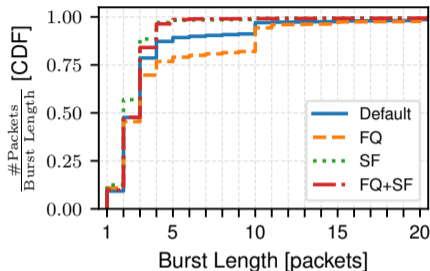


- Fair Queue (FQ) is a queuing discipline (qdisc)
- It can delay sending until a given timestamp

- Fair Queue (FQ) is a queuing discipline (qdisc)
- It can delay sending until a given timestamp
- First observation: FQ makes things worse ?!?!?
- After some investigation:
 - quiche mechanism enters a bad state when losing few packets
 - While in this state, it continuously loses more packets
 - State is only exited if specific ACK is also lost
- We disabled this mechanism in measurements marked with *SF*



- Fair Queue (FQ) is a queuing discipline (qdisc)
- It can delay sending until a given timestamp
- First observation: FQ makes things worse ?!?!?
- After some investigation:
 - quiche mechanism enters a bad state when losing few packets
 - While in this state, it continuously loses more packets
 - State is only exited if specific ACK is also lost
- We disabled this mechanism in measurements marked with *SF*

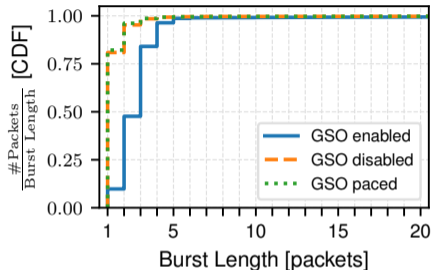


Take-away Results:

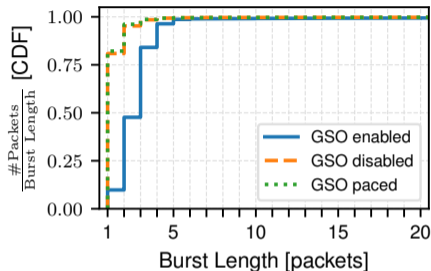
- ACK-clocking leads to good pacing even without FQ
- FQ+SF: smaller bursts → less packet loss → less congestion events → ACK-clocking works better

- Reduces CPU overhead for packet processing
- Creates bursts of packets → influences pacing

- Reduces CPU overhead for packet processing
- Creates bursts of packets → influences pacing
- Without GSO, pacing works better (but: higher CPU load)
- Solution: Pass the pacing rate together with the buffer
- Not implemented in the Linux kernel → kernel patch
- Result: Comparable to *GSO disabled* measurements



- Reduces CPU overhead for packet processing
- Creates bursts of packets → influences pacing
- Without GSO, pacing works better (but: higher CPU load)
- Solution: Pass the pacing rate together with the buffer
- Not implemented in the Linux kernel → kernel patch
- Result: Comparable to *GSO disabled* measurements



Take-away Results:

- GSO influences pacing negatively but reduces CPU load
- The presented kernel patch allows to use GSO without affecting pacing



Take Away Messages

- Pacing behavior differs widely between libraries
- Accurate pacing can be achieved using different approaches
 - Entirely from user-space (picoquic)
 - With the help of kernel functionality, e.g. qdiscs (quiche)
- ACK-clocking already leads to good pacing results
- The presented kernel patch allows to use GSO to benefit from lower CPU load without affecting pacing

More results/details in the paper!



QUIC Steps: Evaluating Pacing Strategies in QUIC Implementations

MARCEL KEMPF, Technical University of Munich, Germany
SIMON TIETZ, Technical University of Munich, Germany
BENEDIKT JÄGER, Technical University of Munich, Germany
JOHANNES SPÄTH, Technical University of Munich, Germany
GEORG CARLE, Technical University of Munich, Germany
JOHANNES ZIRNGIBL, Max Planck Institute for Informatics, Germany

Paper:



<https://doi.org/10.1145/3730985>

Contact:



kempfm@net.in.tum.de



Backup Slides

All CCAs and Implementations

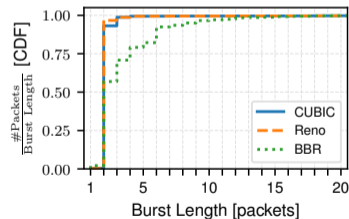
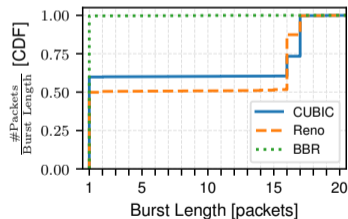
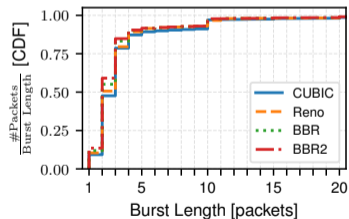
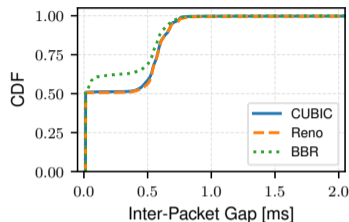
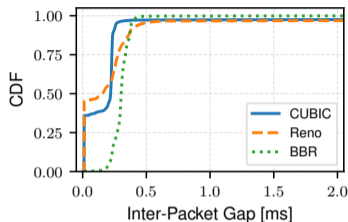
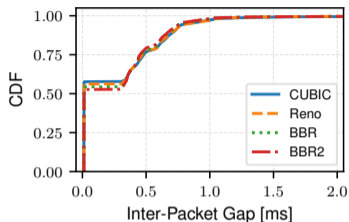


Figure 1: quiche

Figure 2: picoquic

Figure 3: ngtcp2

Why is the threshold 0.1 ms?

- The minimum theoretical inter-packet gap in our setup is approximately 0.012 ms
- 0.1 ms is sufficiently larger to distinguish between fundamental serialization delays and actual bursts or intentionally paced packets