MAY is not enough! QUIC servers SHOULD skip packet numbers

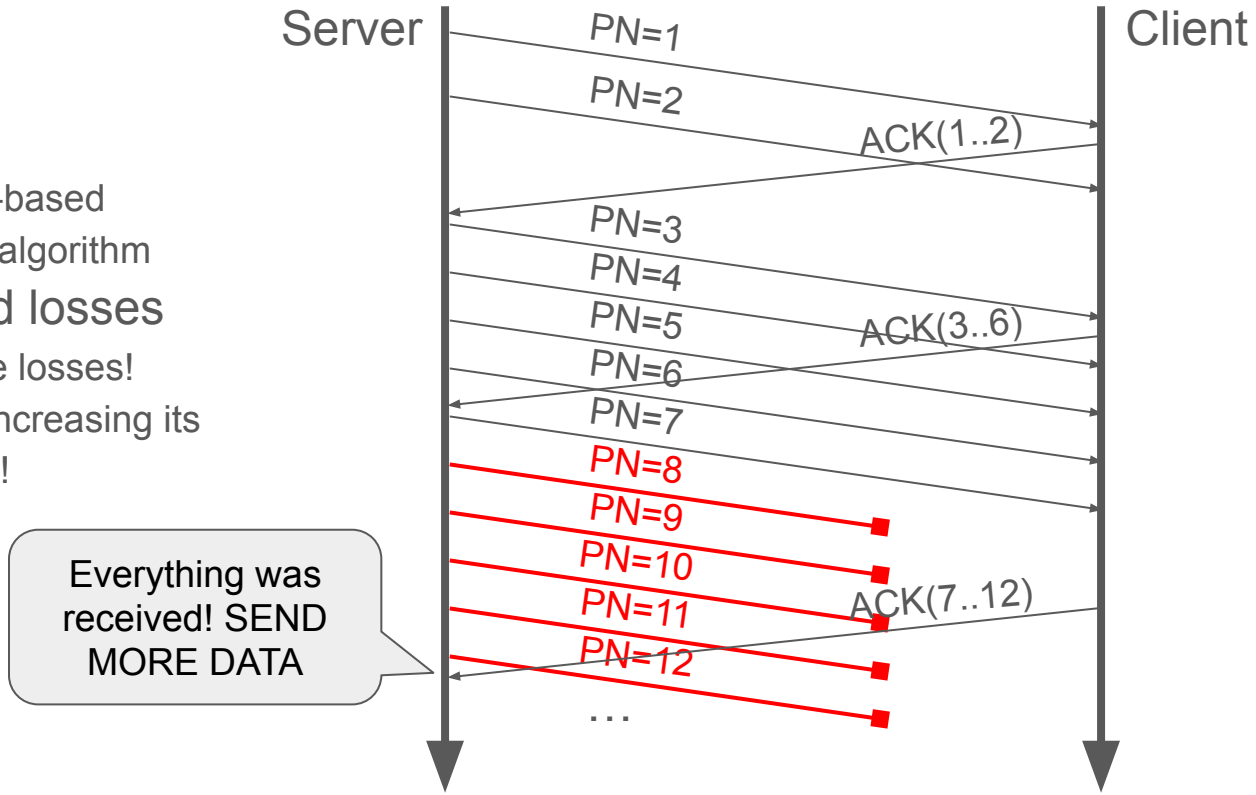
IETF 123 - Madrid, Spain
QUIC

Louis Navarre
Olivier Bonaventure

UCLouvain

Optimistic acknowledgment (OACK) attack: send acknowledgments for packets not (yet) received

- Hypothesis:
  - Server→client flow
  - Malicious client
  - Server uses a loss-based congestion control algorithm
- Congestion-induced losses
  - The client hides the losses!
  - The server keeps increasing its congestion window!



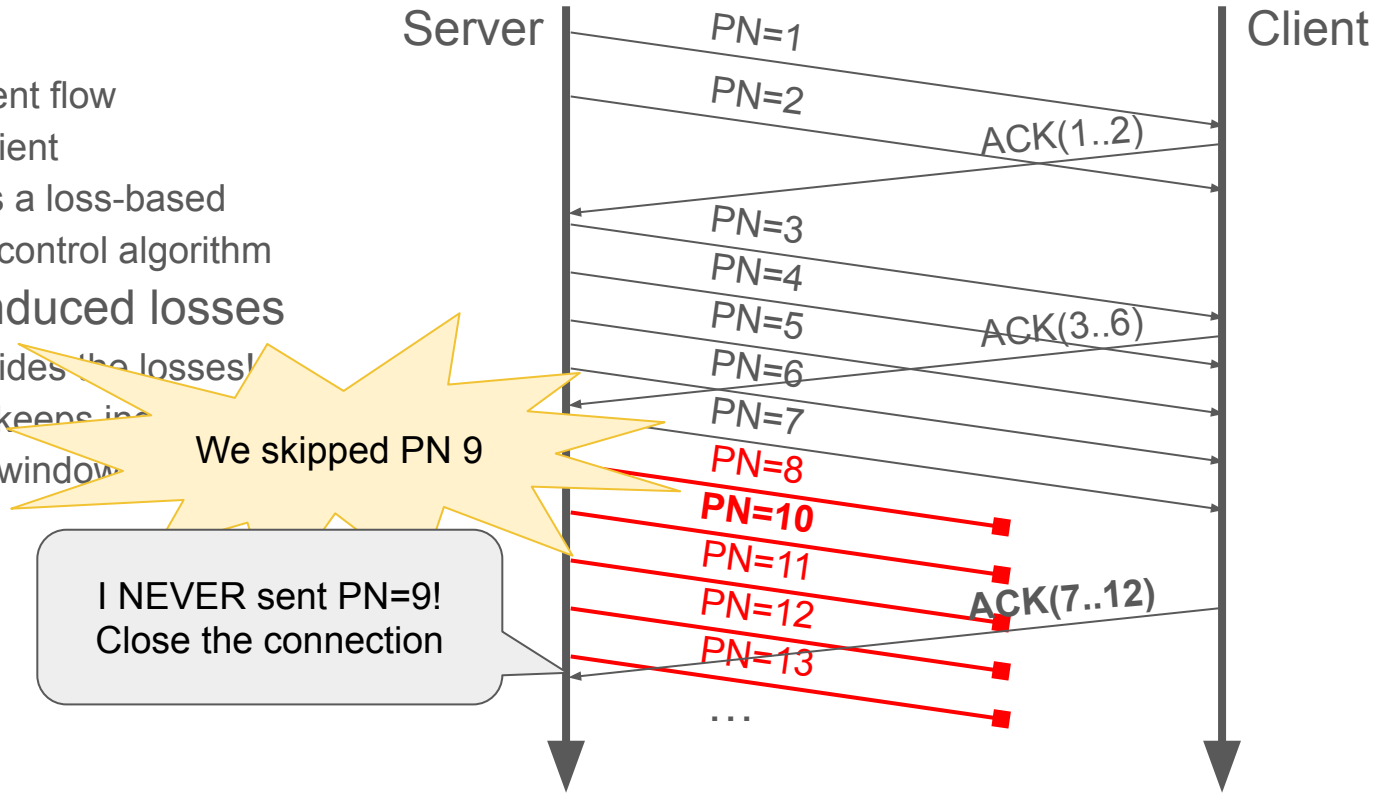
QUIC endpoints MAY skip packet numbers

- What is a packet number? Uniquely identify each QUIC packet
- What is the “Optimistic ACK (OACK) Attack”? Increase the server’s bitrate
- **By skipping packet numbers, QUIC servers detect the OACK attack**

> **21.4. Optimistic ACK Attack**
>
> An endpoint that acknowledges packets it has not received might cause a congestion controller to permit sending at rates beyond what the network supports. An endpoint **MAY** skip packet numbers when sending packets to detect this behavior. An endpoint can then immediately close the connection with a connection error of type PROTOCOL_VIOLATION; see Section 10.2.

Optimistic acknowledgment (OACK) attack: send acknowledgments for packets not (yet) received

- Hypothesis:
  - Server→client flow
  - Malicious client
  - Server uses a loss-based congestion control algorithm
- Congestion-induced losses
  - The client hides the losses!
  - The server keeps increasing its congestion window

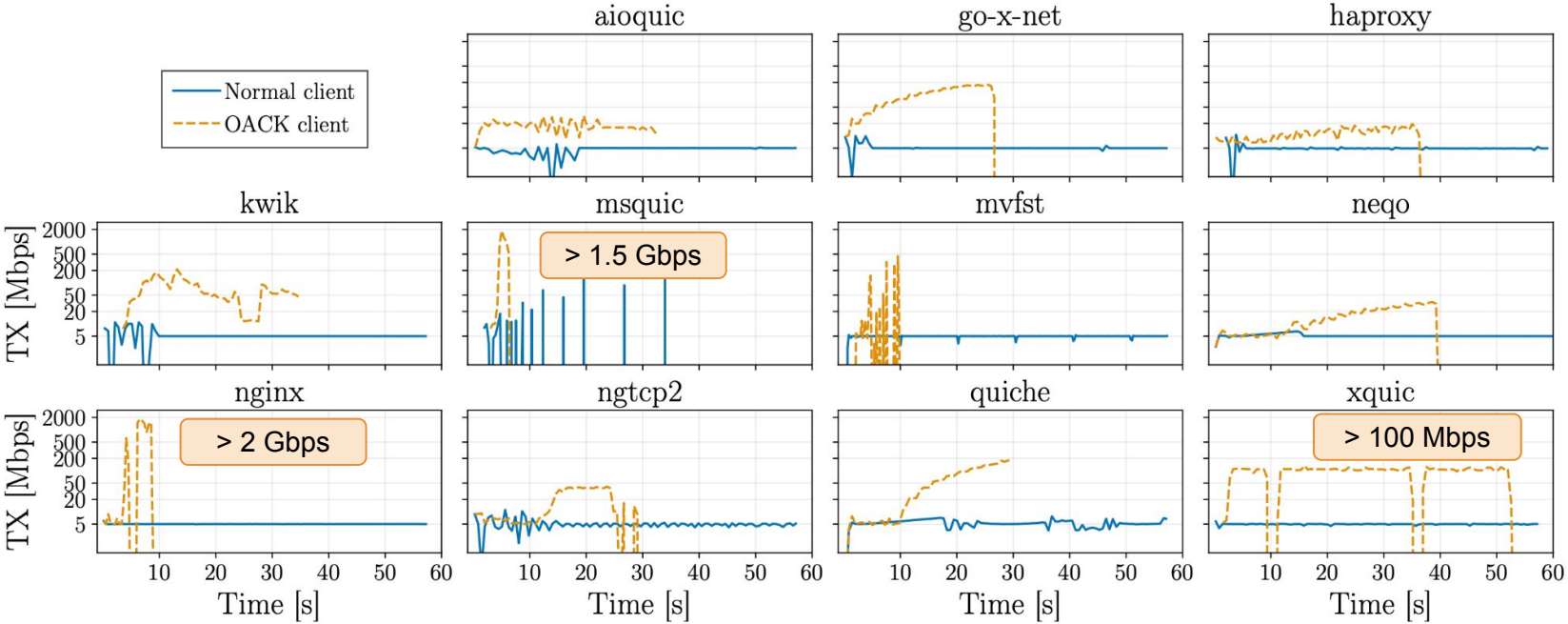


We took the 16 available QUIC implementations from the QUIC Interop Runner

Implem	quic-go v0.50.0	ngtcp2 v.1.11.0	mvfst v2025.03.03	quiche v0.23.4	kwik v0.10.1	picoquic 6304c2e9cc35	aioquic v1.2.0	neqo v0.12.2
(1) Skip PN	✓	✗	✗	✗	✗	✓	✗	✗
(2) Correctness	✓	✓	✓	✗	✗	✓	✗	✓
Implem	**nginx 145b228530c3**	**msquic v2.3.9**	**xquic v1.8.2**	**lsquic v4.2.0**	**haproxy v3.1**	**quinn v0.5.9**	**s2n-quic v1.52.0**	**go-x-net d18fa4cfbd84**
(1) Skip PN	✗	✗	✗	✓	✗	✓	✓	✗
(2) Correctness	✓	✓	✓	✓	✓	✓	✓	✓

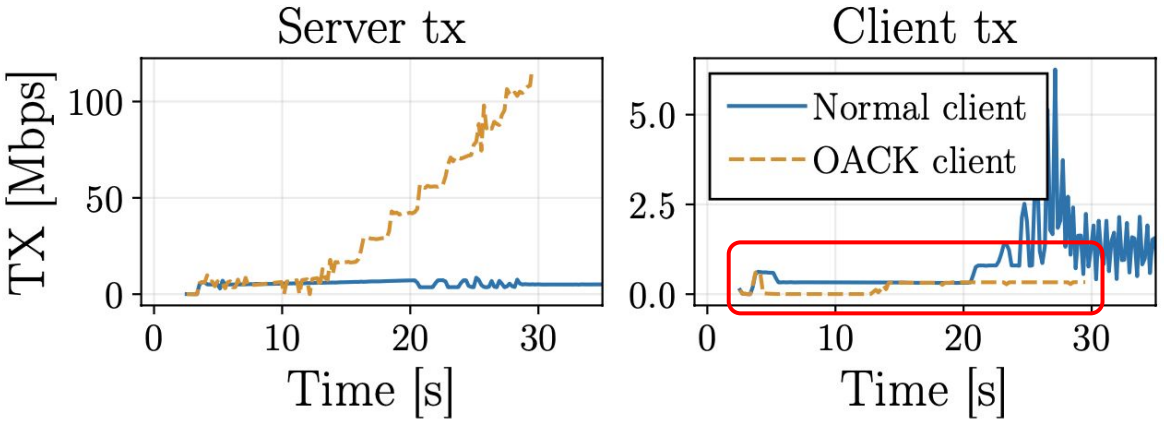
- 11 of them do **not** skip packet numbers
- Hence, vulnerable to the Optimistic ACK attack

The “quality” of the attack varies across implementations



When performing the OACK attack, the client does not show malicious behavior

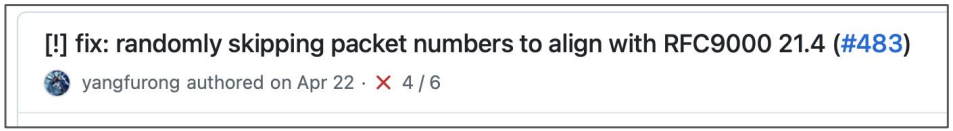
- The content of QUIC packets is end-to-end encrypted
- A middlebox cannot “see” the optimistic acknowledgments
- The middlebox can only see the traffic pattern of the client

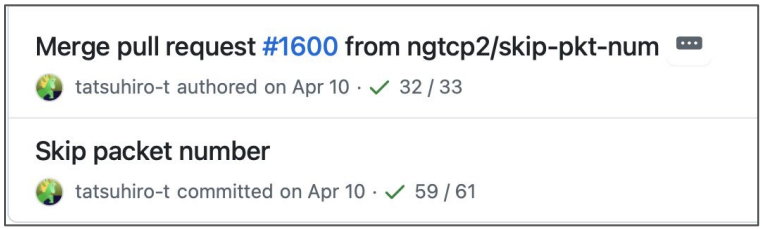


The client does not send more data with OACKs

Discussions with maintainers to prevent the OACK attack

- We contacted the maintainers of all vulnerable implementations
- Patched: go-x-net, ngtcp2, xquic (Alibaba), haproxy, quiche (Cloudflare)
- WIP: mvfst (Meta), nginx
- No answer: msquic (Microsoft)
- Aware but won’t correct it now: neqo (Mozilla), kwik, aioquic





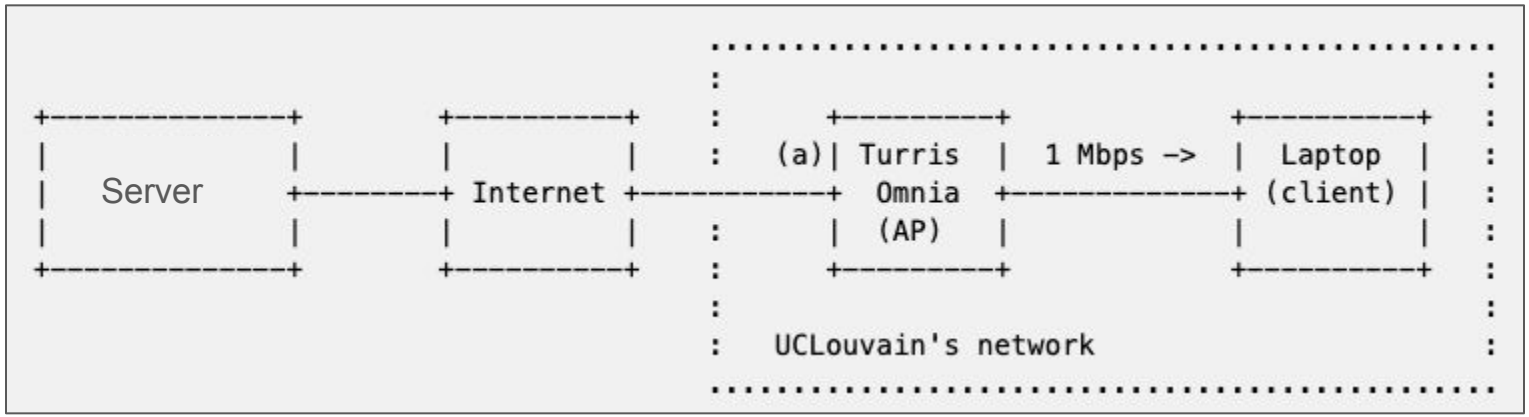
MAY is not enough! QUIC servers SHOULD skip PN

- Skipping packet numbers to prevent the Optimistic ACK attack
- Solution in RFC9000 but not widely implemented
- This work was presented during ANRW 2025
- Published paper (with the same name)
- We focused on server → client flows with malicious client…
- … BUT it is independent of the direction and the application protocol

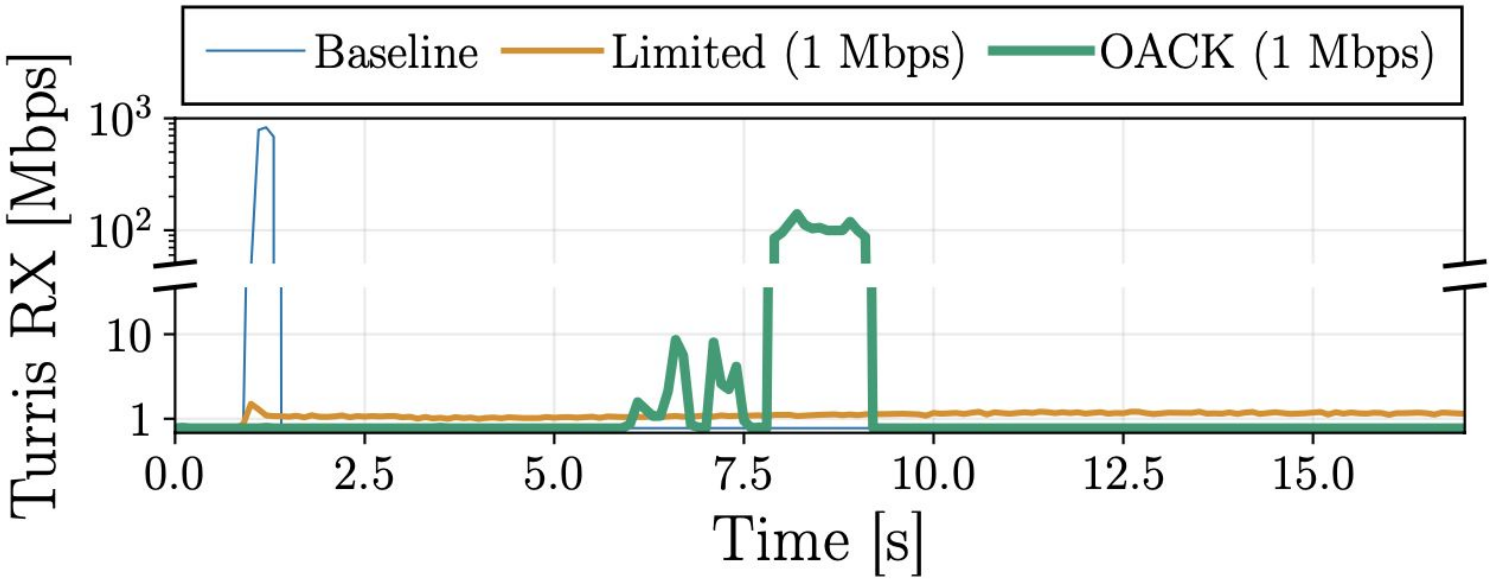
louis.navarre@uclouvain.be
olivier.bonaventure@uclouvain.be

We made real-network experiments “against” a real CDN

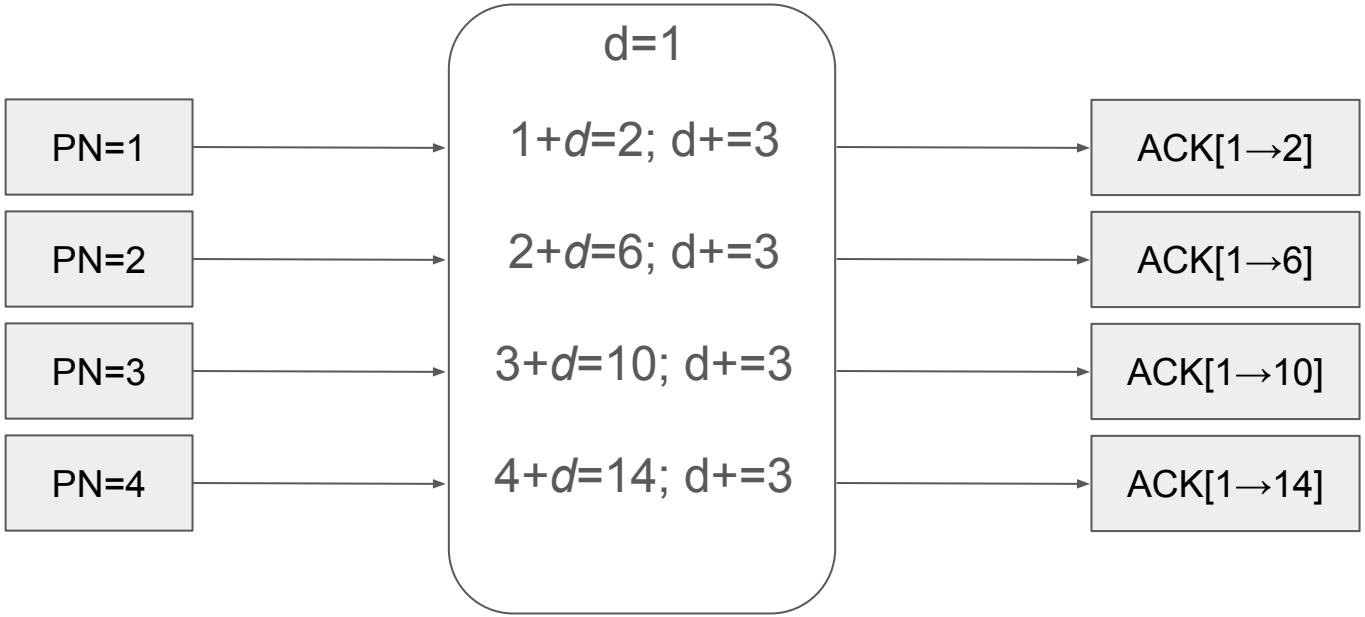
- **With agreement of the CDN**
- We host a website on the CDN to download images
- The CDN provides the network stack, DoS defense, …
- We limit the client’s (RX) throughput to 1 Mbps



We increase by x100 the server bit rate with OACK



ACK predictor: acknowledge *d* more packets in advance and increase *d*



The optimistic acknowledgment is known in TCP since 1999

TCP Congestion Control with a Misbehaving Receiver

Stefan Savage, Neal Cardwell, David Wetherall, and Tom Anderson
Department of Computer Science and Engineering
University of Washington, Seattle

1999

Misbehaving TCP Receivers Can Cause Internet-Wide Congestion Collapse *

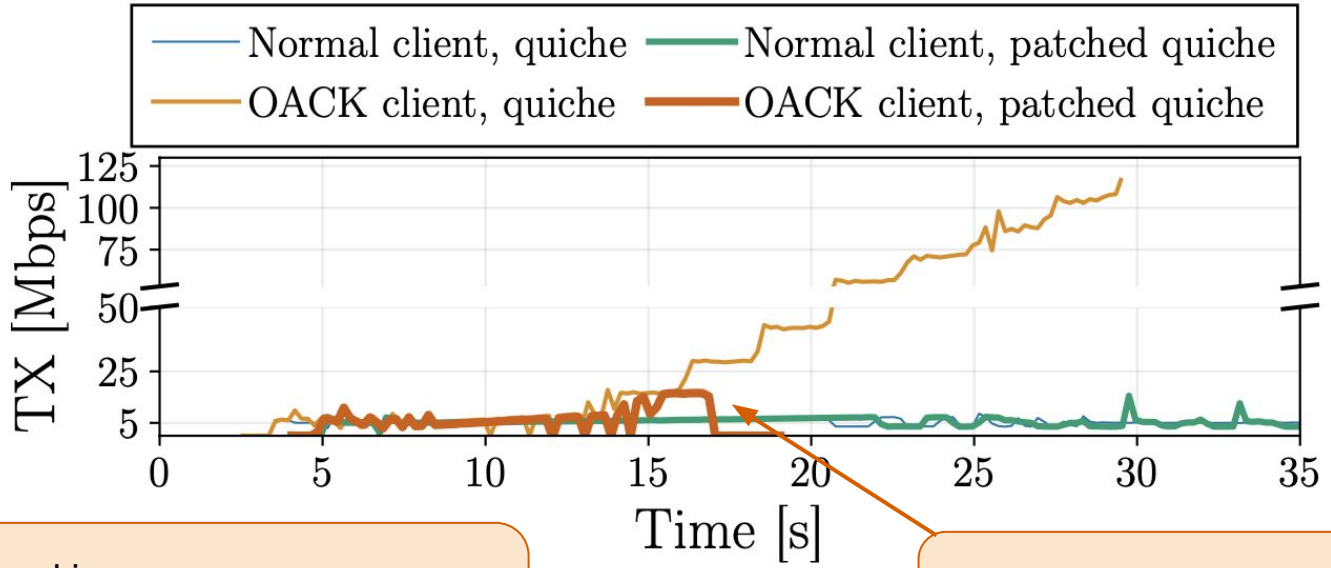
Rob Sherwood
Department of Computer Science
University of Maryland, College Park
capveg@cs.umd.edu

Bobby Bhattacharjee
Department of Computer Science
University of Maryland, College Park
bobby@cs.umd.edu

Ryan Braud
Department of Computer Science and Engineering
University of California, San Diego
rbraud@cs.ucsd.edu

2005

We made a 180-lines patch to quiche to protect against OACK



Here we skip on average one every 1024*128 packets, we **can** be more aggressive to detect it more quickly

The server receives an ACK for a skipped packet number

The optimistic ACK attack in TCP: well, yes, but actually, no

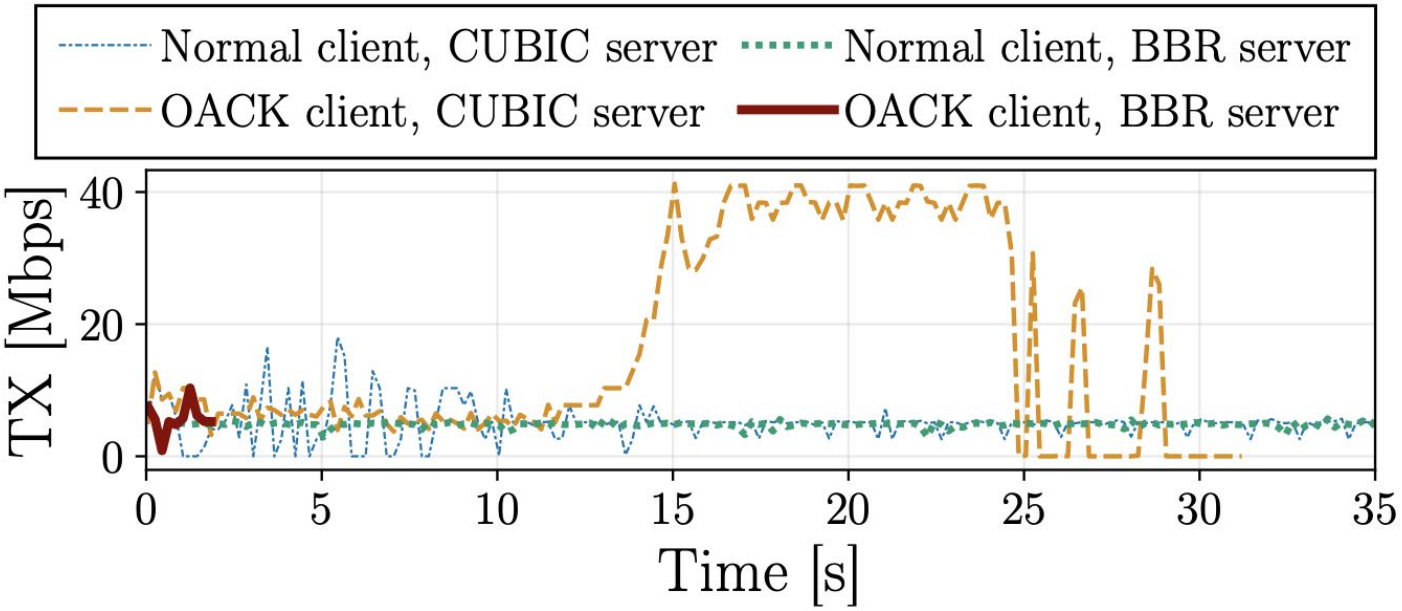
- Suggested fixes in the past, no wide adoption
- Ongoing discussions at the TCPM meeting this IETF!

Discuss with that guy (from our team)!

(There are more stickers on its laptop now)

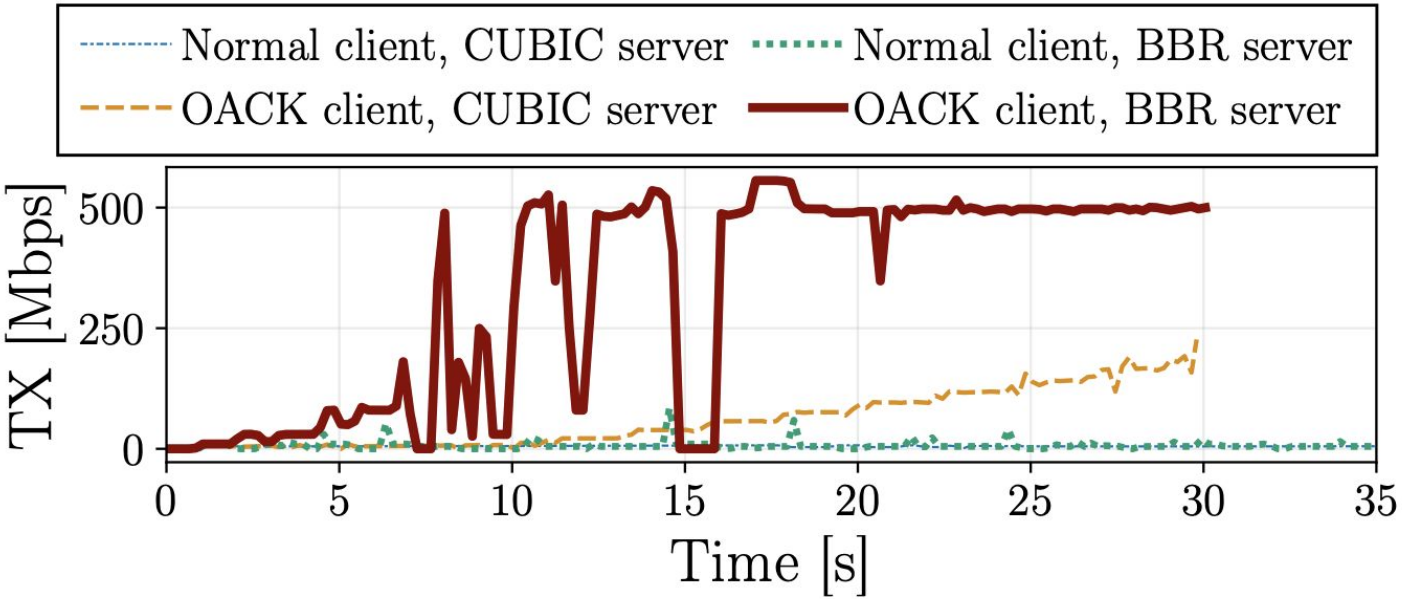
BBR is “less” predictable than CUBIC

- When the target implementation **does** check the ACK correctness



BBR is “less” predictable than CUBIC

- When the target implementation **does not** check the ACK correctness



Our simple optimistic acknowledgment predictor

- Send an ACK frame for every new received packet (benign behavior)
- Acknowledge *d* more packets than the packet number received
- Continuously increase the value of *d*
- Delay the attack (attacking from the start is delicate)

Algorithm 1: Simple QUIC OACK predictor.

i_d: aggressiveness of increase of d
l: number of packets before starting OACK
d: OACK increase. Init: 0
n_m: maximum received packet number
n_0: first packet number with data. Init: None
n: received packet number
Result: Potential OACK ranges to send to the peer

```
1 if n_0 is None then
2 |   n_0 ← n
3 end
4 if n_0 + l < n or n < n_m then
5 |   return No range;
6 end
7 n_m ← n;
8 d ← d + i_d;
9 return n_0..n + d;
```