

Opportunistic ACKs in TCP

Matthieu Baerts (presenter)

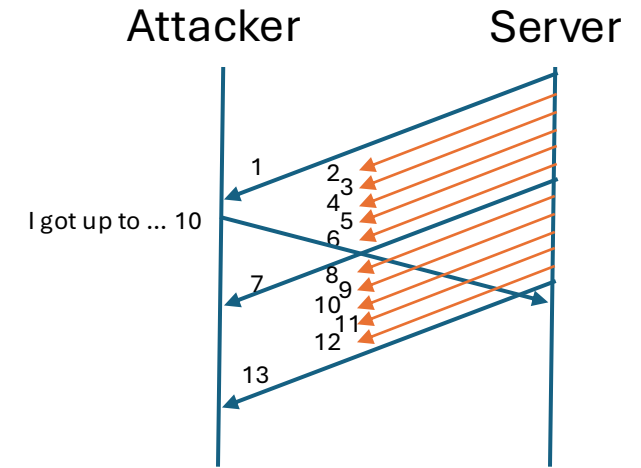
Louis Navarre

Olivier Bonaventure

UCLouvain & WELRI , Belgium

Problem

- A misbehaving receiver (attacker) could send ACKs for not-yet-received data.
- That pushes the sender to send at a higher throughput than expected.
- Already documented in 1999:
 - Savage, S., Cardwell, N., Wetherall, D., & Anderson, T. (1999). TCP congestion control with a misbehaving receiver. ACM SIGCOMM Computer Communication Review, 29(5), 71-78.



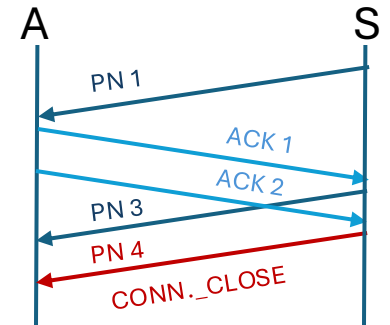
Why looking at OACK now?

MAY is not enough!
QUIC servers SHOULD skip packet numbers

Louis Navarre
louis.navarre@uclouvain.be
UCLouvain, Belgium

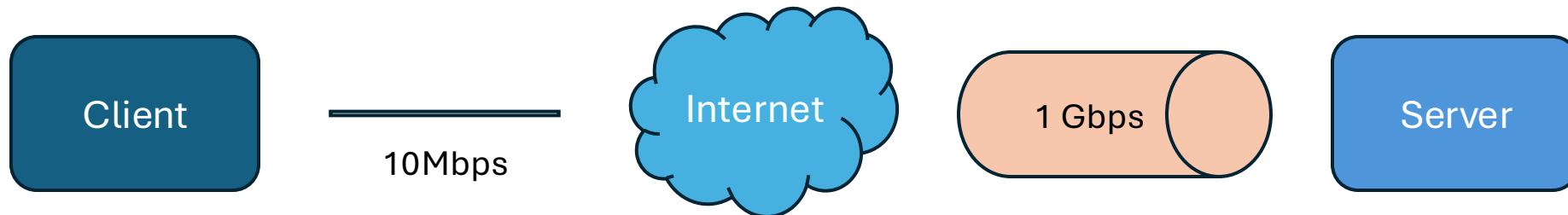
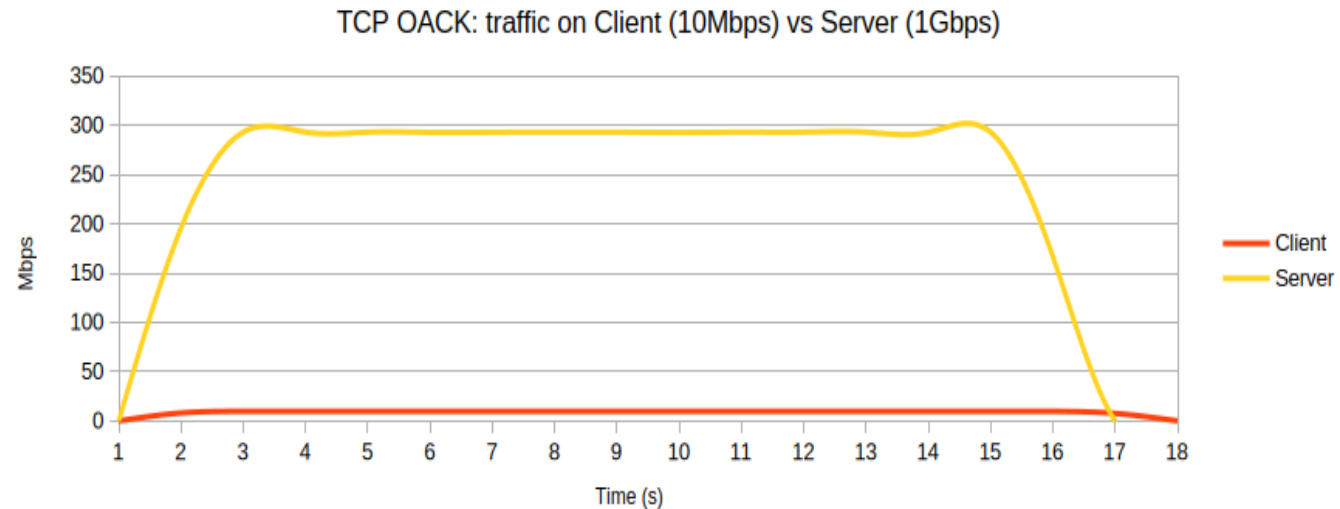
Olivier Bonaventure
olivier.bonaventure@uclouvain.be
UCLouvain & WEL-RI, Belgium

- Similar issue on QUIC side: ANRW25
- QUIC has a solution: sender can skip packet numbers
- Most QUIC server implementations now support the solution
- But what about TCP stacks?



Linux TCP and opportunistic acks

- Opportunistic ACKs attack still works: →
- If cautious, the attack can be undetectable: ACKs can be "simply" sent at a higher rate



Detecting such attacks

- Observing ACK for unseen data (SKB drop tracepoint on Linux)
- Can be difficult... but some hints:
 - Limit to active flows with large congestion window and one direction
 - No retransmissions, low ACK vs sent segments ratio, low RTT variation
 - Most of them can be faked, or it depends on the attack: no clear hints that an attack is in place

Possible Mitigations

- Not changing the TCP protocol, e.g.
 - New TCP option, or behaviour change (i.e. using QUIC's counter-measure)
 - Refusing ACK for "too large" segments
- Not impacting the performances, e.g.
 - Stop and wait for a feedback
- Not relying on other layers, e.g.
 - TLS heartbeat

Counter-measure: suggestion

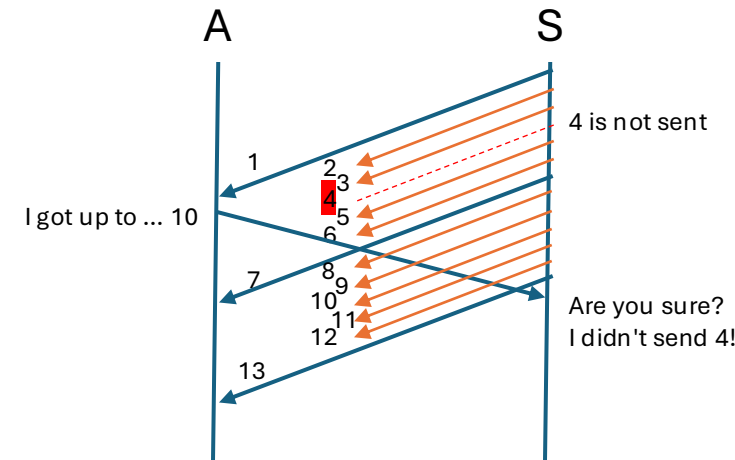
- Randomly "drop" a segment in long downloads

- e.g. the 4th one on the graph →

- If this segment gets ACKed by the client: attack detected

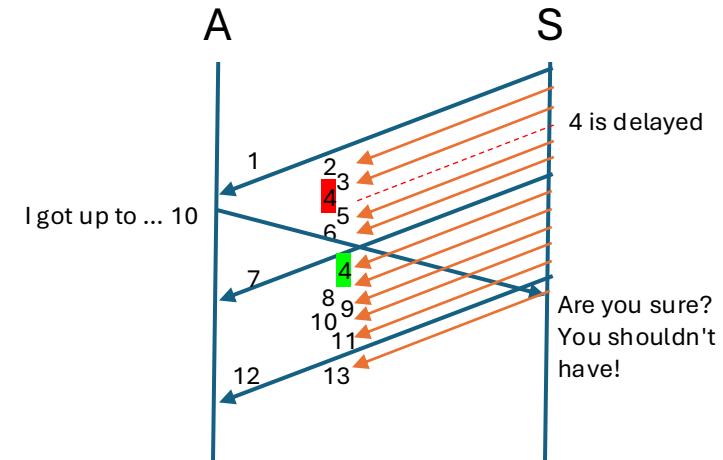
- If server receives dupacks or SACKs: OK, (re)transmit

- It has a cost... perf, latency, buffering, etc.



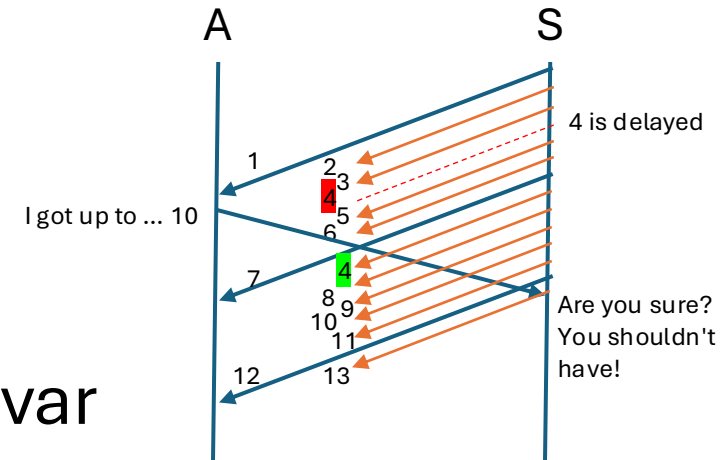
Counter-measure: optimisations

- Do not tell TCP congestion control algorithms when the *skipped* segment is (re)transmitted.
- Delay the retransmission by $>$ half an RTT:
 - Instead of waiting for dupacks / SACKs before retransmitting
 - Might be more complex to detect an attack



Counter-measure: conditions

- Large congestion window + active + one direction
- No retransmissions + low ACK/Data ratio + low rttvar
 - (Can be faked, depending on the attack)
- Excluding some prefixes: LAN, private addresses
- Can be easily disabled
- More?



Counter-measure: simple alternatives

- Limit throughput per connection:
 - `SO_MAX_PACING_RATE` on Linux
- Slow down the connection in some conditions:
 - When too many "large" segments are ACKed

Counter-measure: or nothing?

- Not a new concept: no solutions today, why?
 - Not causing bad troubles? Or not exploited so far?
 - Is it easier today to have a "large" botnet than smaller doing opportunistic ACKs?
 - Or simpler counter-measure (max pacing) enough? Even for smaller actors?
 - Or simply ... not seen because the egress is not that monitored?
- What should then be done?