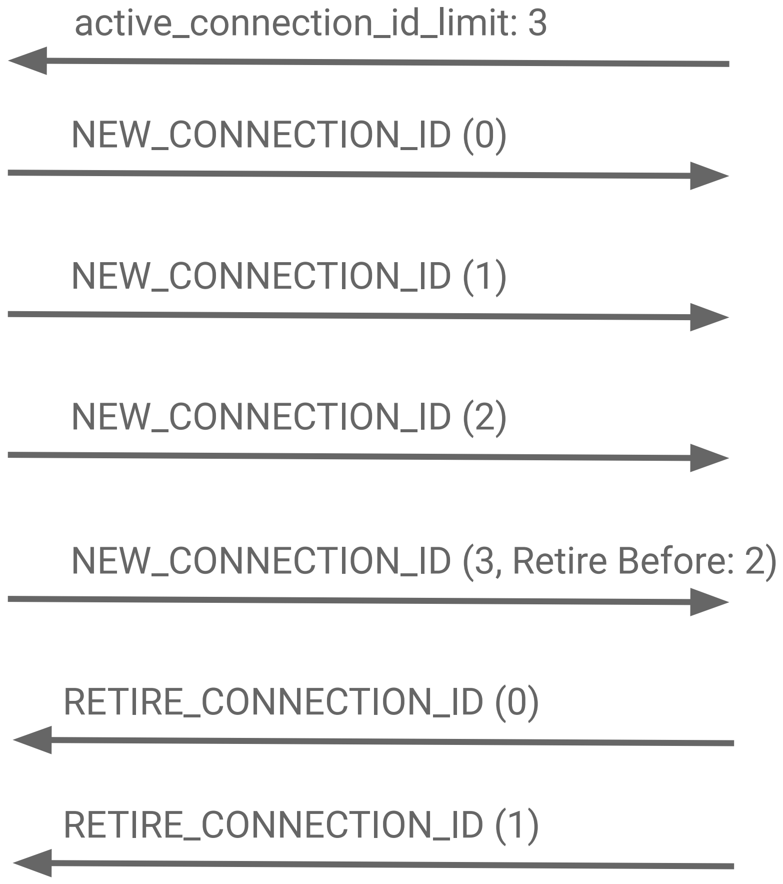
QUIC Resource Exhaustion Attacks

Marten Seemann, IETF 119

QUIC Connection ID Flow Control

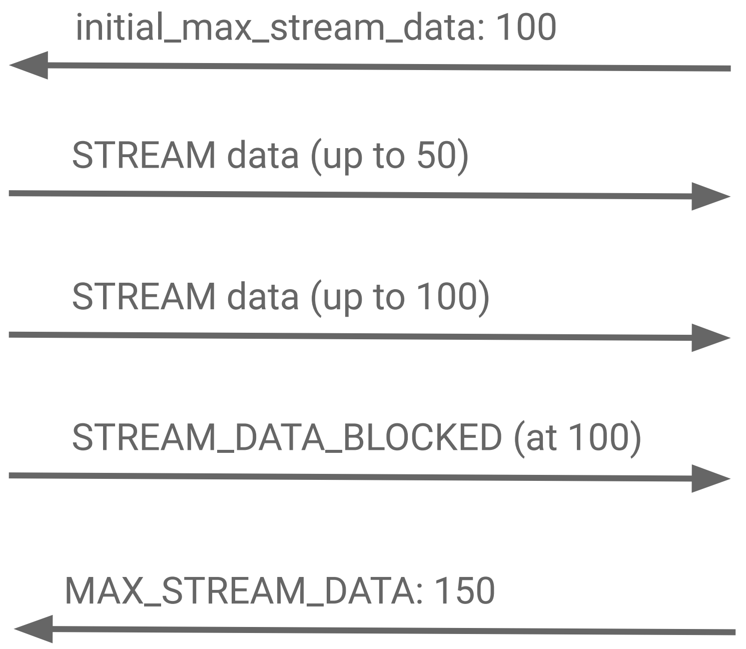


Congestion Control

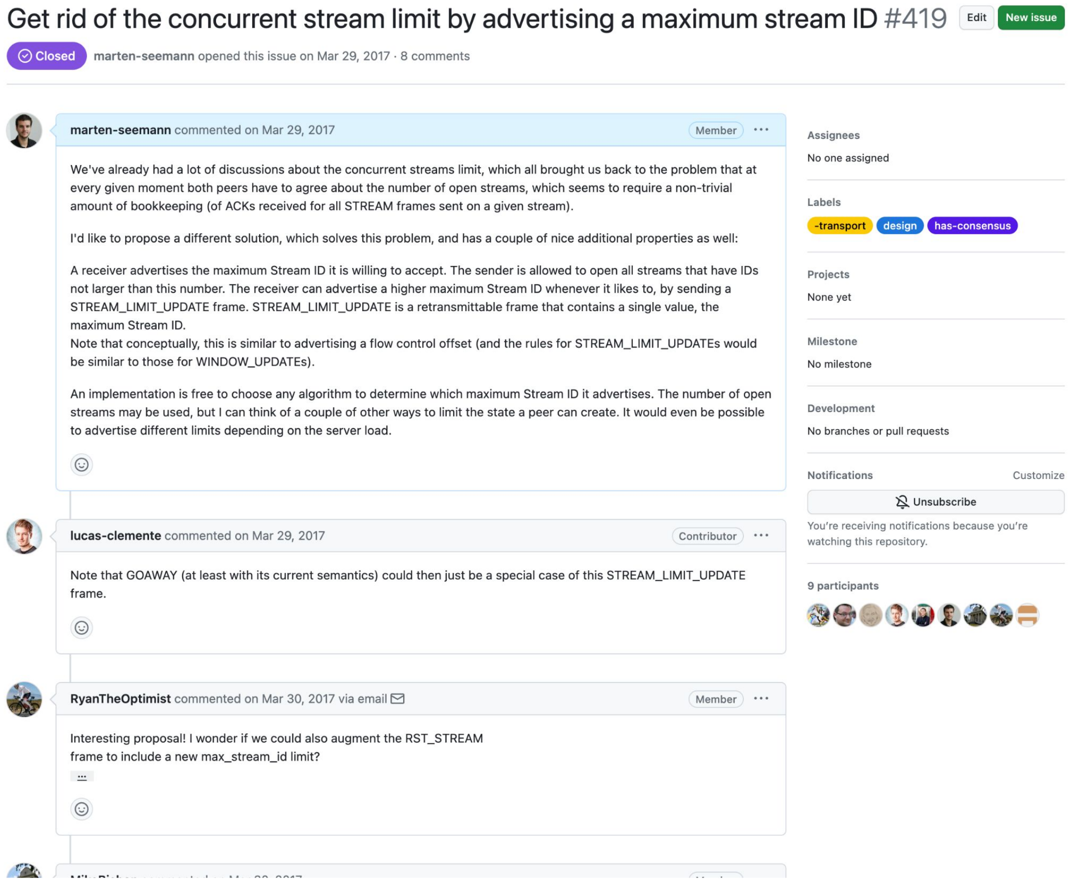
- on packet loss: reduce the congestion window
- on repeated packet loss: minimum congestion window is 2 (full-size) packets
- RTT measurement can be inflated by the peer

Compare to:

QUIC Stream Flow Control



Similar to HTTP/2's Rapid Reset Attack



Get rid of the concurrent stream limit by advertising a maximum stream ID #419

Closed · marten-seemann opened this issue on Mar 29, 2017 · 8 comments

marten-seemann commented on Mar 29, 2017 · Member

We've already had a lot of discussions about the concurrent streams limit, which all brought us back to the problem that at every given moment both peers have to agree about the number of open streams, which seems to require a non-trivial amount of bookkeeping (of ACKs received for all STREAM frames sent on a given stream).

I'd like to propose a different solution, which solves this problem, and has a couple of nice additional properties as well:

A receiver advertises the maximum Stream ID it is willing to accept. The sender is allowed to open all streams that have IDs not larger than this number. The receiver can advertise a higher maximum Stream ID whenever it likes to, by sending a STREAM_LIMIT_UPDATE frame. STREAM_LIMIT_UPDATE is a retransmittable frame that contains a single value, the maximum Stream ID.
Note that conceptually, this is similar to advertising a flow control offset (and the rules for STREAM_LIMIT_UPDATEs would be similar to those for WINDOW_UPDATEs).

An implementation is free to choose any algorithm to determine which maximum Stream ID it advertises. The number of open streams may be used, but I can think of a couple of other ways to limit the state a peer can create. It would even be possible to advertise different limits depending on the server load.

lucas-clemente commented on Mar 29, 2017 · Contributor

Note that GOAWAY (at least with its current semantics) could then just be a special case of this STREAM_LIMIT_UPDATE frame.

RyanTheOptimist commented on Mar 30, 2017 via email · Member

Interesting proposal! I wonder if we could also augment the RST_STREAM frame to include a new max_stream_id limit?

...

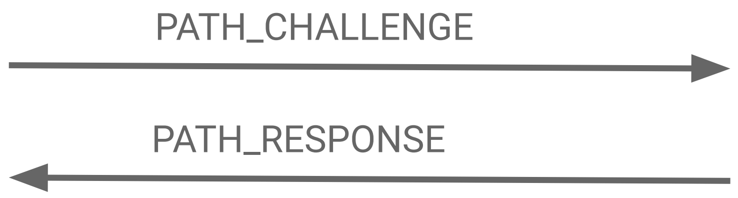
Better Flow Control for Connection IDs?

Should we have introduced a MAX_CONNECTION_ID frame?

For now, limiting the number of RETIRE_CONNECTION_ID frames mitigates the attack.

https://seemann.io/posts/2024-03-19-exploiting-quics-connection-id-management/

Path Validation is vulnerable, too



https://seemann.io/posts/2023-12-18-exploiting-quics-path-validation/