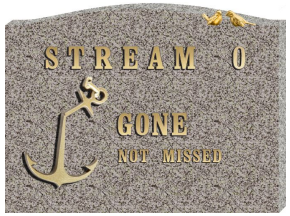
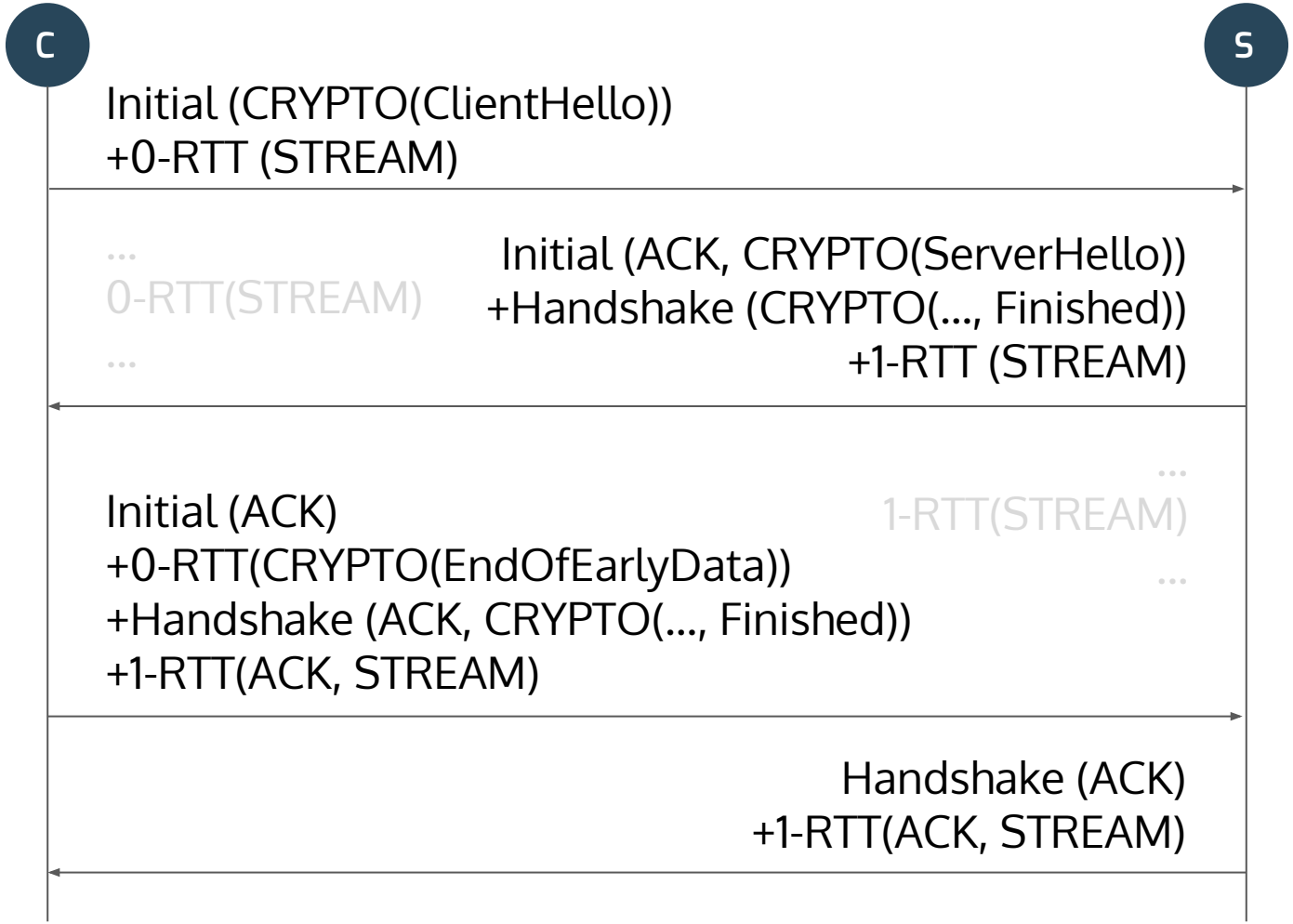
Mo' Problems

QUIC IETF 102, Montreal, July 2018

Martin Thomson

When Can Keys Be Destroyed? (#1544)





Simple Solution: Timers

Treat each packet number space separately

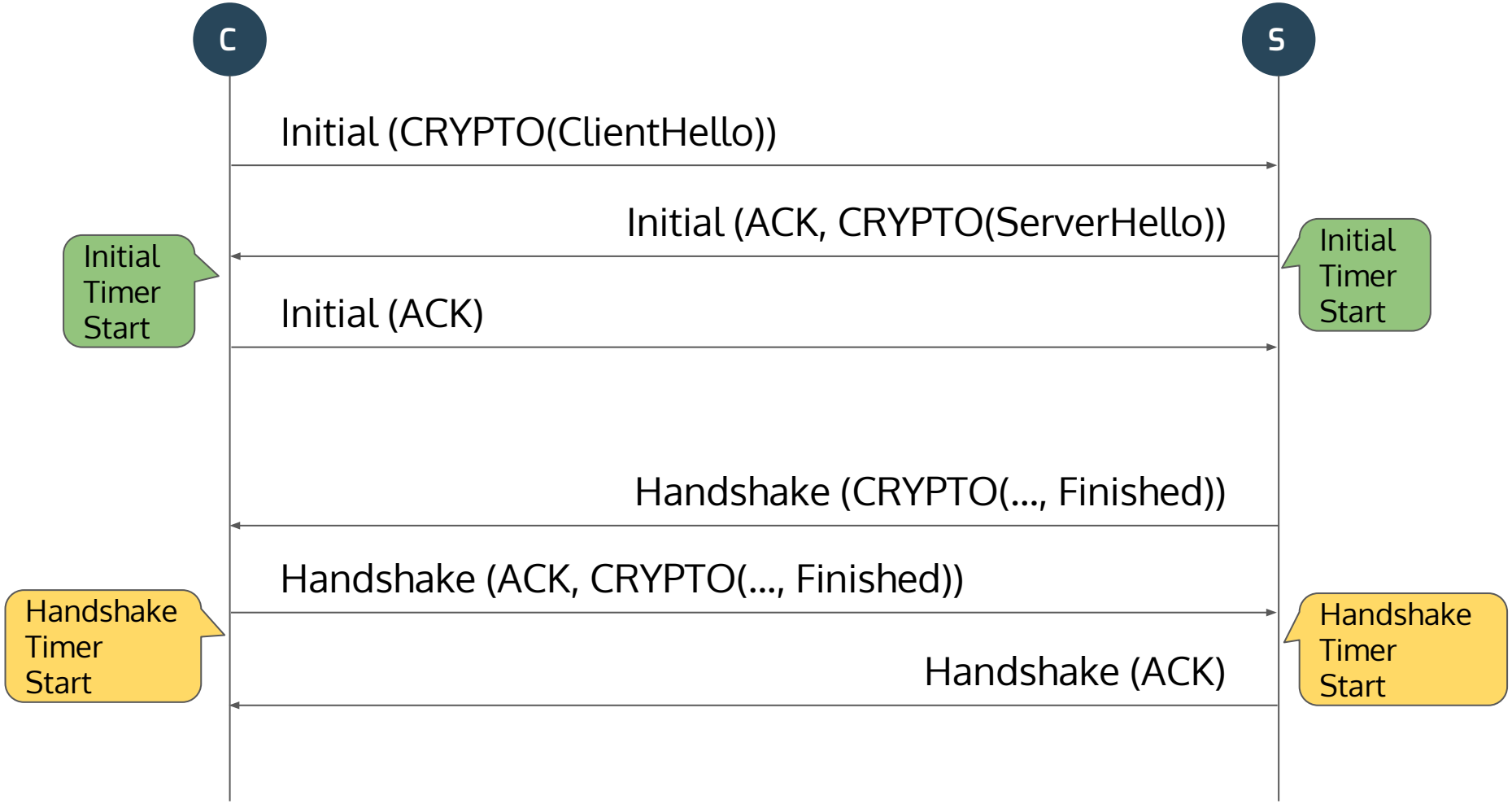
A space is done when both read and write keys for the next space are ready

Set a timer when done and destroy the keys when it expires

- ... until then, resend CRYPTO and send ACK as normal
- ... afterwards, drop packets protected with those keys

The timer can be long-ish (no practical harm in infinite)

Separate Packet Number Spaces



Optimization: Implicit Acknowledgment

Receiving Handshake packets implies that all CRYPTO frames from Initial packets was received

Receiving 1-RTT packets at a server means that all CRYPTO frames in Handshake packets was received by the client

Receiving acknowledgments for 1-RTT packets at a client means that all CRYPTO frames in Handshake and 0-RTT packets was received by a server

Stop sending those CRYPTO frames then

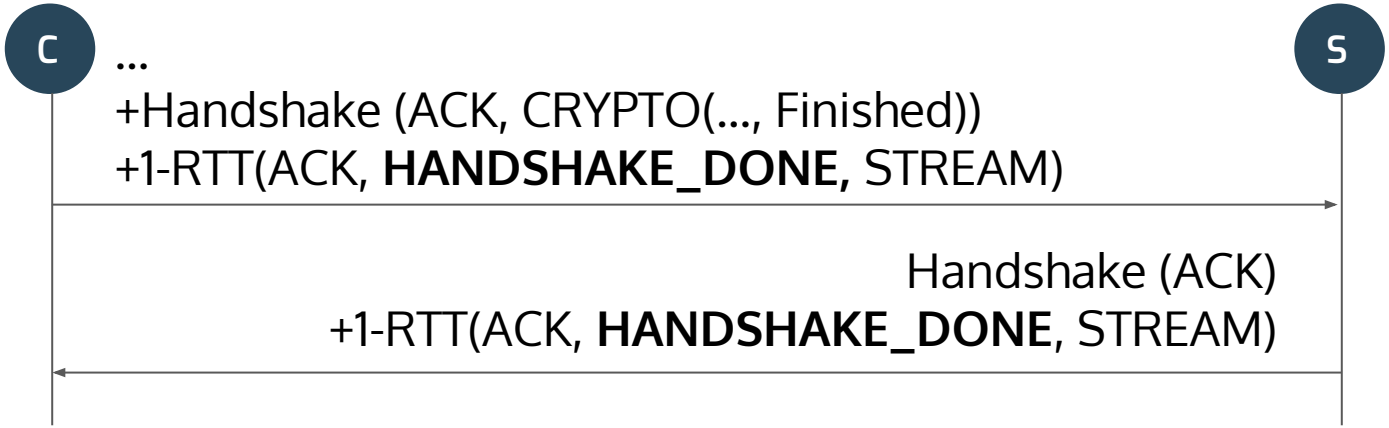
Let the packets with ACK frames that appear afterwards drop

Alternative: HANDSHAKE_DONE Frame

An explicit signal that an endpoint believes that the handshake is done

Retransmitted until acknowledged

On receipt endpoints could destroy all handshake keys



Proposal: Document Timer-based Cleanup

Optimizations are fun, but they don’t need to be standard

Key Updates

How do we signal this?

TLS method (unilateral KeyUpdate message) doesn't work

DTLS method (acknowledged KeyUpdate) requires visibility of ACKs

QUIC method (KEY_PHASE signaling) permits trial decryption

Maybe couple updates to connection ID changes

Consider our options here