1 Status Quo

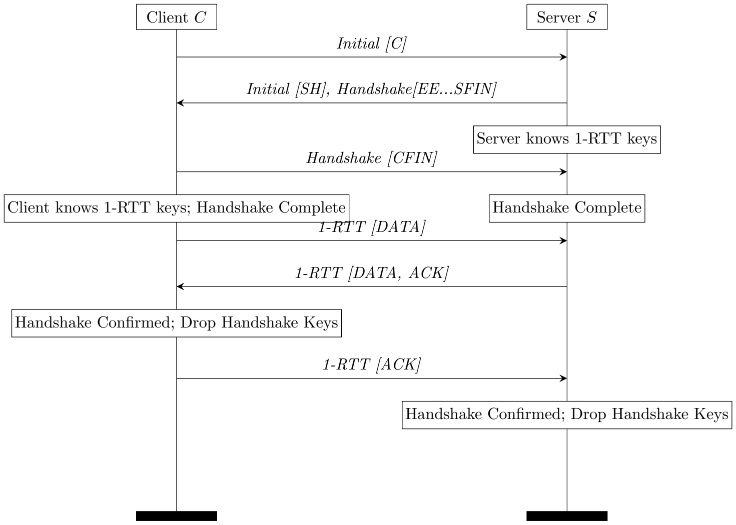
Client *C*
Server *S*
Initial [C]
Initial [SH], Handshake[EE...SFIN]
Server knows 1-RTT keys
Handshake [CFIN]
Client knows 1-RTT keys; Handshake Complete
Handshake Complete
1-RTT [DATA]
1-RTT [DATA, ACK]
Handshake Confirmed; Drop Handshake Keys
1-RTT [ACK]
Handshake Confirmed; Drop Handshake Keys

Figure 1: Basic QUIC handshake (without 0.5 RTT data).

Figure 1 shows the basic QUIC handshake along with the various checkpoints. This just for reference and hopefully non-controversial.

2 Never Drop the Keys (PR 3121)

The basic idea here is to (1) never drop the Handshake keys and (2) continue to send ACKs for packets you have received (as agreed upon separately in YUL, you MUST ACK each packet at least once). The result is that the handshake goes quiescent even if no application data is ever transmitted, though you will not reach “handshake confirmed” in this case, as shown in Figure 2.

One obvious question is what happens in the case where you *are* sending data. In the basic case, we would just expect ACK piggybacking on the data, as shown in Figure 3

You can also safely implement implicit ACK. At the point where you have handshake confirmed, you can stop sending any of your Handshake DATA frames (though you still need to ACK any of the peer’s Handshake frames, or you will get deadlocks if the peer doesn’t implement implicit ACKs.)

Kazuho had raised two concerns about this design:

1. Because handshake confirmed is not guaranteed, key update is confusing.
2. It is possible to have outstanding handshake data at the point where you want to do migration.

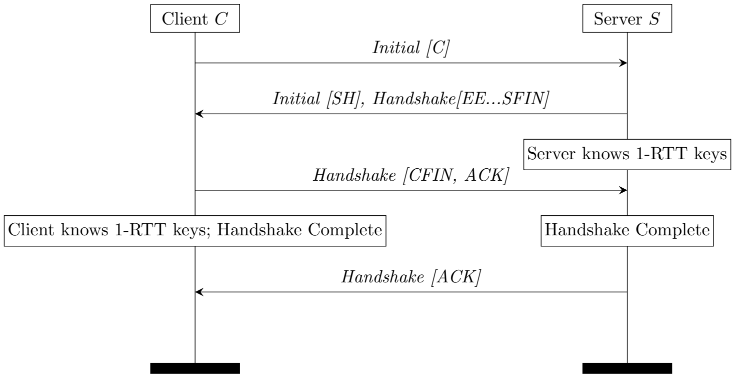

Figure 2: Never drop keys (no data sent)

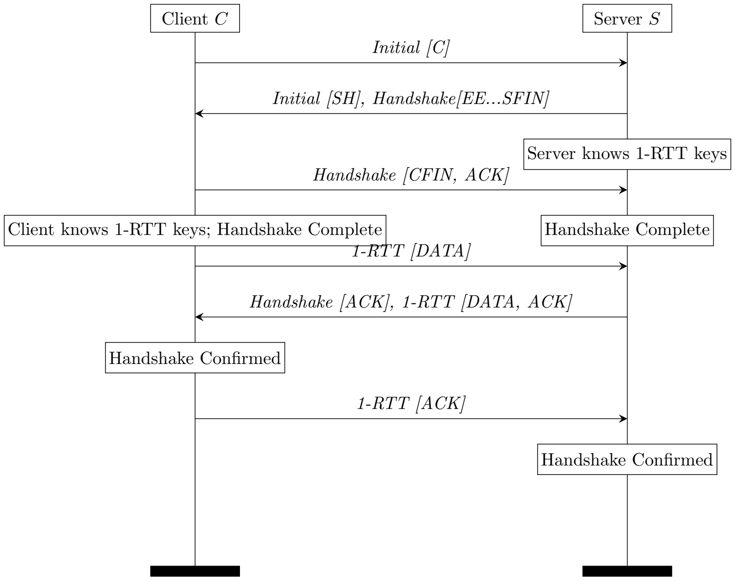

Figure 3: Never drop keys (data sent)

I don't really understand point (1). As noted in `https://github.com/quicwg/base-drafts/issues/3212`, the handshake confirmed test for key update is redundant, and so we can just re-code this as "don't initiate key update for key $n + 1$ until you are sure the other side has key n". And this just lets us have the usual rule without a special case for the handshake → 1-RTT transition.

Point (2) needs some elaboration. As a general matter, performing migration while any handshake data is outstanding is a recipe for bridging the two paths. In the current document this is addressed by (1) requiring that the client not initiate migration prior to handshake confirmed and (2) requiring that the