Discarding QUIC Old 1-RTT Keys Design Team Proposal

David Schinazi, Google

Design Team Members
Eric Rescorla
Martin Duke
Marten Seemann
Martin Thomson
Kazuho Oku
Christian Huitema

Problem Statement

QUIC short headers contain KEY_PHASE bit

Allows for unilateral key updates without prior permission requests

If endpoint updates keys twice without peer knowing,
can end up disagreeing on current key epoch is

Design Principles

Avoid trial decryption

Explicit signal to agree on new epoch before updating again

Not driven by acknowledgments or special retransmission logic

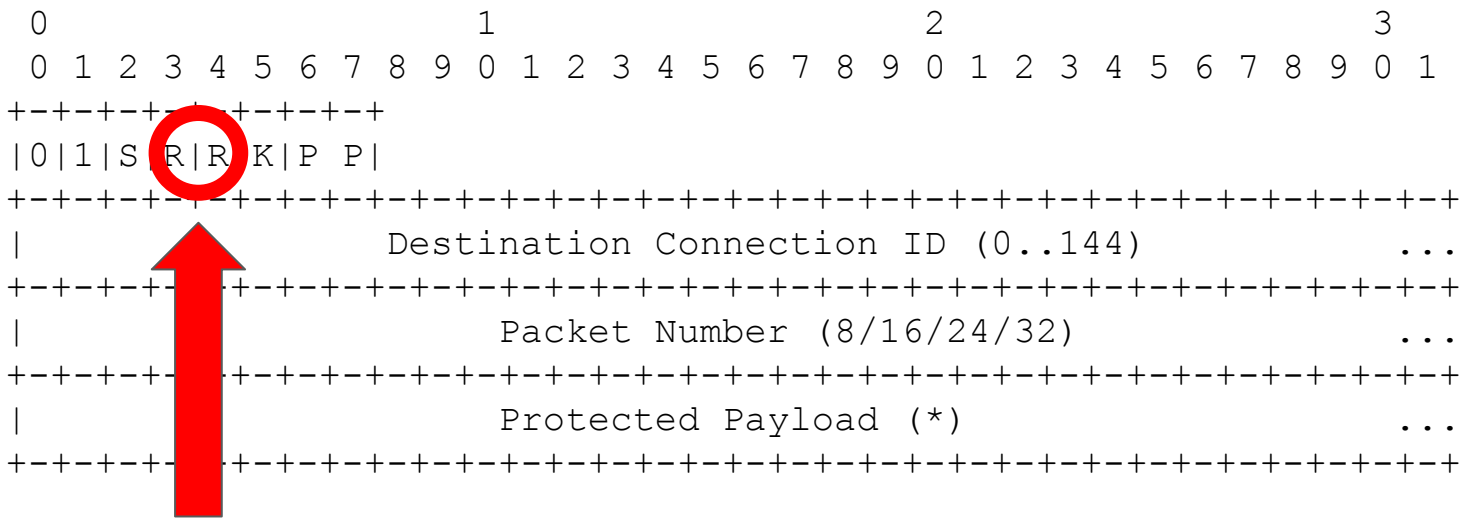
Model: endpoints unilaterally initiate update then require confirmation

Endpoints can update their send keys and force peer to update send keys

Simple implementations but need to support two 1-RTT read keys

Proposal

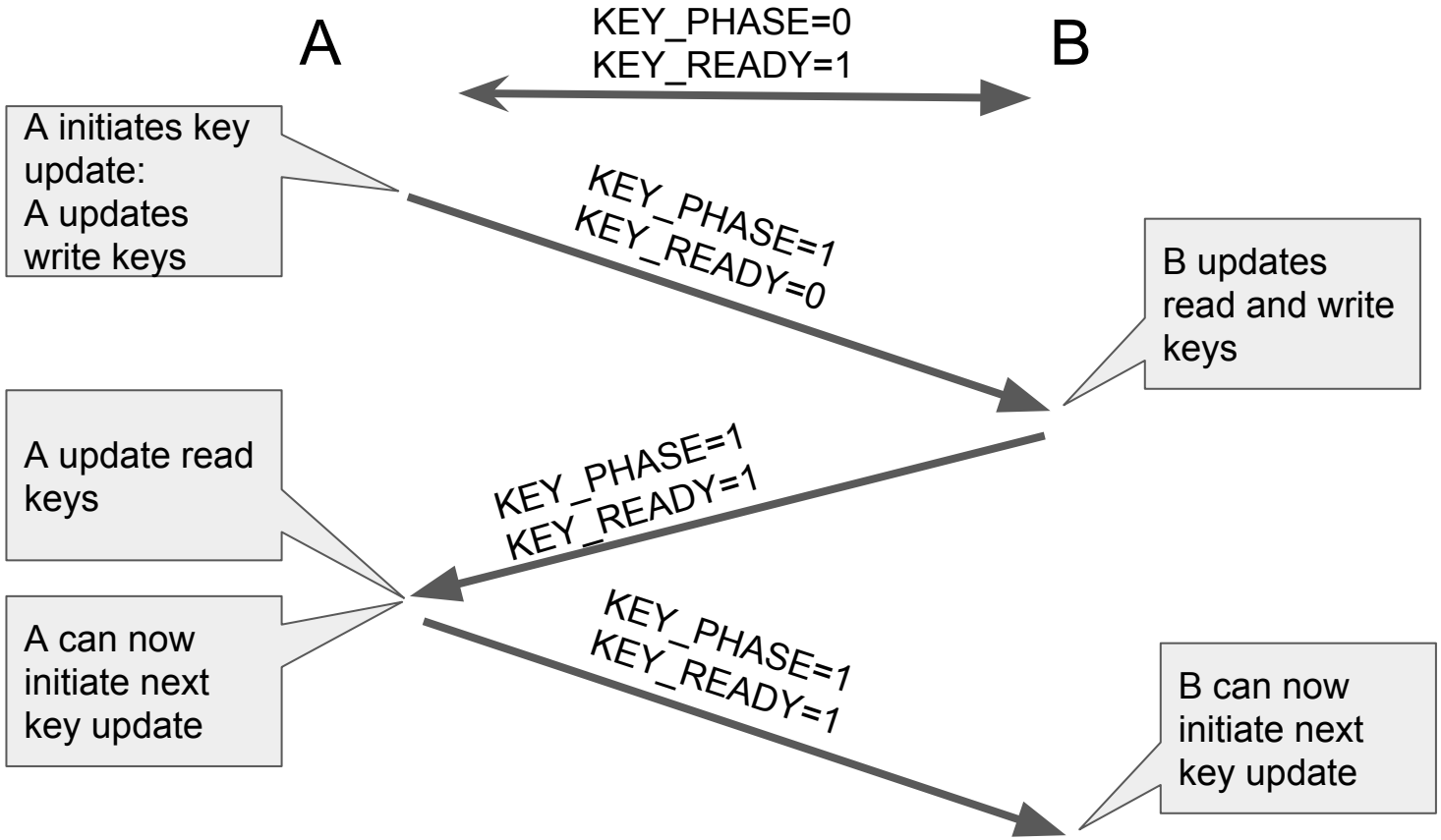
New encrypted bit in short header: KEY_READY



KEY_READY Bit

Send KEY_READY at a given key phase after you've received at given key phase

Do not initiate key update until after receiving KEY_READY


A
KEY_PHASE=0
KEY_READY=1
B
A initiates key update:
A updates write keys
KEY_PHASE=1
KEY_READY=0
B updates read and write keys
A update read keys
KEY_PHASE=1
KEY_READY=1
A can now initiate next key update
KEY_PHASE=1
KEY_READY=1
B can now initiate next key update

TODO: Limit Excessive Key Updates

Problem: if
- A initiates key updates as soon as possible
- B only keeps two keys in memory
- There is packet reordering

Then: valid packets dropped — performance degradation

Solutions:
- B waits before sending KEY_READY
- A waits before initiating next key update
- Accept that excessive key updates harm performance

Consequences minor, please send opinions to list

Discarding QUIC Old 1-RTT Keys Design Team Proposal

David Schinazi, Google

Design Team Members
Eric Rescorla
Martin Duke
Marten Seemann
Martin Thomson
Kazuho Oku
Christian Huitema

(backup slide: simultaneous key update still works)