Discarding Old Keys

QUIC, IETF 104, Prague, March 2019
Martin Thomson, David Schinazi, Kazuho Oku, Marten Seeman, <your name here>
Goals

As discussed in the Tokyo interim:
Discard Initial keys as soon as possible
Discard Handshake keys when appropriate
Signal when a key update can be initiated
Use explicit signals rather than implicit ones, or timers
Basic Idea

Use an explicit signal for all key transitions

Initial -> Handshake

Handshake (+ 0-RTT) -> 1-RTT

$1\text{-RTT}_n \rightarrow 1\text{-RTT}_{n+1}$

The signal indicates when it is safe to discard old keys
Options

KEYS_READY #2237
RETIRE_KEYS #2492
MAX_KEY_UPDATES #2504

A bit in the first octet (old version of #2237)
KEYS_READY

KEYS_READY is sent when read keys are available

... when the peer is expected to use corresponding keys

Implicitly identifies keys

Initiator of a key update has to suppress old frames

When sent and received

older keys can be discarded
and new key updates initiated
KEYS_READY Key Update

Can update again from here

Can update again from here
RETIRE_KEYS

RETIRE_KEYS send when no more data will be sent

- Initial->Handshake = first packet (special case for server)
- Handshake->1-RTT = after all data is acknowledged
- Key Update = sent when new keys installed

Implicitly identifies keys

RETIRE_KEYS is retransmitted until acknowledged

When both sent and received, old keys can be discarded

Subsequent key updates can be initiated once received and sent has been acknowledged
RETIRE_KEYS Key Update

I

M
N
N
N

ACK

Can update again from here

R

M
N
N
N

ACK

Can update again from here

Can update again from here
MAX_KEY_UPDATES

Cap key updates rather than control discarding of keys

Explicit counter in frame sets cap on updates

Fixes key update issues, limited fix for handshake:

No signal for Initial->Handshake transition
Handshake->1-RTT signaled with MAX_KEY_UPDATES=0
First frame is sent after all Handshake data ack’d
MAX_KEY_UPDATES

CRYPTO
ClientHello

Initial

0-RTT

CRYPTO
ServerHello

Initial

Hands hake

1-RTT

Hands hake

1-RTT

Implicit drop of Initial Keys

MAX_KEY_UPDATES
(0)

CRYPTO
... Finished

Implicit drop of Initial Keys

ACK
Use a bit instead

Same basic semantics as the frame (ideally KEYS_READY)

Carried in every packet

No special retransmission rules
Common characteristics

Use a frame (explicit signal agreed in Tokyo)

An endpoint can block key updates by not sending the frame

Both KEYS_READY and MAX_KEY_UPDATES allow a 3PTO delay to cap active read keys at an endpoint to 2

The time limit is aspirational, as no mechanism exists to force an endpoint to send the proposed frames
Difference: Explicit vs. Ambient Signal

Explicit: counter in frame

- Drawbacks: octets, allows for >1 update

Ambient: use the encryption level

- Drawbacks: need to suppress any retransmission when initiating a key update
Initial -> Handshake Transition

MAX_KEY_UPDATES says that the implicit signal is OK
The other proposals use an explicit signal

Table. Initial keys are dropped when receiving ...

<table>
<thead>
<tr>
<th></th>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYS_READY</td>
<td>Handshake(KEYS_READY)</td>
<td>Handshake(KEYS_READY)</td>
</tr>
<tr>
<td>RETIRE_KEYS</td>
<td>Handshake(RETIRE_KEYS)</td>
<td>Handshake(RETIRE_KEYS)</td>
</tr>
<tr>
<td>MAX_KEY_UPDATES</td>
<td>Initial(ServerHello)</td>
<td>any Handshake packet</td>
</tr>
</tbody>
</table>
Trigger

KEYS_READY - matching read keys available

RETIRE_KEYS

Handshake: all data from previous epoch acknowledged exception for server: immediately

1-RTT: when all CRYPTO data is acknowledged

Update: send immediately, no update until acknowledged

MAX_KEY_UPDATES - trigger isn’t important