Stream 0, Architecture, Layering, and DTLS

Eric Rescorla
ekr@rtfm.com
Agenda

- The current architecture (brief)
- A brief survey of stream 0 problems
- Root cause analysis
- "Crypto Stream"
- Layering
Current Layering Architecture

- TLS Handshake
- Streams
- QUIC Packets
- Keys/State
- Application
“Stream 0” Problems (Data)

- Partly encrypted, partly not
  - Retransmission
  - Boundary between SFIN and NST
- Very tight coupling with the TLS stack
  - Boundaries between flights
  - Is this an SH or an HRR (or a stateful versus stateless HRR go)
- Exempt from flow control during the handshake but not later
  - You can go negative
- Mismatch between QUIC and TLS 1.3 notions of 0-RTT boundaries
- Can’t bundle unencrypted and encrypted in one packet
- QUIC sure knows a lot about crypto
  - + Double encryption
“Stream 0” Problems (ACKs)

- Complicated ACK rules
  - Just a pain to reason about and implement
- Holes from unencrypted packets being ACKed in enc packets
- Contradictions between ACKs and handshake state
  - SFIN means CFIN received but might not contain ACKs
- Reliability for the CFIN
“Stream 0” Problems (code)

- Constantly special cased in people’s code

```c
uint32_t
StreamPair::ResetInbound()
{
    // this is used in a very peculiar circumstance after HRR on stream 0 only
    assert(mStreamID == 0);
    return mIn->ResetInbound();
}
```
What's the source of the problem?

- We're to set up a reliable transport
- The reliable transport depends on keys which come from TLS
- But TLS requires a reliable transport to work
Breaking the dependency cycle

- Step 1: Separate the transport used by the crypto from the transport used by the application
- Step 2: ???
- Step 3: Profit
Crypto Streams and Crypto ACKs

Diagram:

- TLS Handshake
- Application
- Crypto Streams
- Crypto ACKs
- Streams
- QUIC Packets

Keys/State
Crypto Streams Implications

- Probably the minimal change that does anything
- Solves some of the problems
  - Flow control
  - Clarity about what’s encrypted and what’s not (at cost of widening the TLS interface further)
  - Holes from unencrypted packets
  - ACK rules
- Need some solutions for the other problems
  - HANDSHAKE_DONE for CFIN?
  - Or live with them....
Crypto Streams and Crypto ACKs

Keys/State

<table>
<thead>
<tr>
<th>TLS Handshake</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Streams</td>
<td>Crypto ACKs</td>
</tr>
<tr>
<td></td>
<td>Streams</td>
</tr>
<tr>
<td></td>
<td>QUIC Packets</td>
</tr>
</tbody>
</table>
Layer over DTLS

- DTLS Handshake
- Application
- QUIC Transport (Mostly Streams)
- DTLS Records
DTLS Implications

- **Bigger change to QUIC**
  - Exempts crypto from flow control entirely (reverts previous decision)
  - Mostly deletions!

- **Some small changes to DTLS 1.3**
  - Mostly stealing QUIC packet formats and connection ID structure (which came from DTLS actually)

- **Solves effectively all these problems**
DTLS Impact (I)

- **Bigger change to QUIC**
  - But mostly deletions!
- **Small changes to DTLS 1.3**
  - Primarily importing features from SIP (more later)
  - Conveniently the I-D is still open
- **Solves effectively all these problems**
- **Implementation experience shows a significant net simplification in the QUIC code**
DTLS Impact (II)

- DTLS becomes the QUIC wire image
  - We have flexibility because the I-D isn’t done
  - Ultimately could mostly graft QUIC packet formats onto DTLS 1.3
- Small amount of packet expansion (maybe?)
- ACKs require QUIC having
  - access to DTLS packet #s
  - Epochs require changing ACKs a bit
    - Something like this is also required by crypto streams
- DTLS 1.3 spec and implementations less mature than TLS 1.3
Discussion

What is the right architecture for QUIC?

How do we evaluate the alternatives?
Backup slides
An Alternative for the Schedule Sensitive

DTLS Handshake

Application

Ready

QUIC
(Mostly Streams)

DTLS Records
QUIC Packets
Alternative

- (Somewhat) bigger change to QUIC
  - But mostly deletions!
- Solves effectively all these problems
- Challenges
  - Need to write a document describing how to carry DTLS data over QUIC records
    - Straightforward mapping to DTLS records
  - Small amount of packet expansion (maybe?)
  - DTLS epochs require changing ACKs a bit
    - Though this is also required by crypto streams
  - DTLS implementations less mature