Coding for QUIC

draft-swett-nwcrg-coding-for-quic-00

Ian Swett

Marie-José Montpetit

Vincent Roca

IETF 101 London

Adding Coding to QUIC

- Top level requirements
  - Must not change QUIC v1
    - Use proposed(PR#1072) extension mechanism to negotiate
  - Agnostic WRT the code
    - Can be a block or sliding window code to be negotiated
    - More than one code could be available to a QUIC session
  - Coding takes place within a stream (reusing existing header fields plus a new frame type) or potentially across a few streams
    - This is motivated by the fact that not all streams need to be coded
    - Control frames are typically not as latency sensitive
  - Coding is end-to-end within encryption (like QUIC)
    - Re-encoding only possible with trusted middleboxes
  - Coding happens before encryption
    - Coding does not interfere with encryption

data -> encoding -> encryption

To Code or not to Code

- Some streams maybe coded, some not
- Coding negotiated in QUIC handshake
- One or more coding extensions are offered, allowing 1 or more to be negotiated
  - Final decision on which to use based on application or operational decisions

Framing

- New QUIC frame is defined
  - type: Repair symbol with coding type
  - stream ID: Stream ID being repaired
  - offset: The first source symbol in the window
  - data length: total bytes of coding

Extension:

Repeated Stream ID and offset

Coding Symbols (1)

- Original idea: QUIC packets numbers
  - Packets are lost, so protect that unit
- But:
  - Coding can’t change QUIC Packet Numbers
  - Want to allow
    - Non-consecutive packet protection
    - “Holes” in the sequence not due to losses
      - ie: Path migration
  - Could exceed MTU when adding coding overhead
  - Multipath makes it more complex

Coding Symbols (2)

- New idea (update to the draft):
  - Use an extension frame that references one or more streams

- Only protects latency sensitive data
- Re-uses existing stream send and receive buffers to recover.

Coding Symbols (3)

- New(er) idea:
  - Define an extension frame that replaces a Stream with coded data.

- Allows any type of code.
- Avoids interaction with QUIC's retransmission based recovery.
- Allows maximum flexibility during experimentation.

Next Steps (1)

- Finalize the formatting/initial design:
  - Use QUIC's extension mechanism.
  - PR#1072
- Choose a sample code
  - Raptor one option
  - RS is already open source
- Implement in picoquic?

Next Steps (2)

- Agree on an API to allow different codes to be used without large code changes.
- draft-roca-nwcrg-generic-fec-api-01?

Next Steps (3)

- Make this a RG item?
  - Or
- Migrate to QUIC WG?
  - Or
- Wait for experimentation?

QUESTIONS?

lanswett@google.com
marie@mjmontpetit.com
vincent.roca@inria.fr