

Multipath in Chromium

(not in Chromium anymore)

Presented by Fan Yang

High-level Design

- Designed for gQUIC, not IETF QUIC
- PathID identifies the path
 - gQUIC didn't have multiple connection IDs; PathID was in packet header
- Unified ACK frame acknowledges packets received on all paths
 - Effectively multiple packet number spaces
- Separate congestion controller and loss detection per path
 - Retransmissions could go over a different path than original

Implementation

Design wasn't that difficult, but implementation...

Retransmissions were **very** complex*

Implementation maintained packet buffers instead of stream buffers

Data structure at sender maintained sent packets with data within them

Once packet was sent, moving its data to another path was complex

(Implementation has changed since:

sender now maintains data in stream form, simplifying this immensely)

If your code structure is not conducive, implementation will be hard

Scheduling

Driven by application's primary goals for multipath use ...

Latency minimizing, bandwidth maximizing, reliability, etc.

... and by the properties of the deployment environment

Costs: monetary, battery, etc.

Never got enough buy-in from a customer to help develop a scheduler

“Can you only send the GET on every path?”

“Why don't you improve connection migration first and see if it's enough?”

Lessons Learned

Multipath increases code complexity

This can be small or huge, depending upon the implementation

Scheduling is hard and depends on the use case

If you're not working closely with an application,
you're likely to fail

Most use cases only need connection migration

Once connection migration worked well, no one asked for multipath