
MULTIEPLEX
MULTIPLEX
Multiplexing
in the year 2024

Disclaimer and prior art

These slides are a respin of work or topics that have been discussed at the IETF and the wider HTTP community. Including:

- Slides from Kazuho Oku and Lucas Pardue presented at IETF 119
- Side meetings or hallway conversations with no attribution
- Comments in various other channels

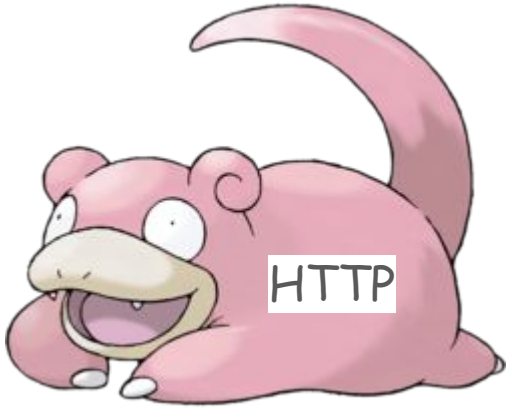
There may be views expressed here that don't fully represent any individual's opinions

HTTP Multiplexing Uses (and Abuses)

- Single-stream context
  - HTTP request/response concurrency
  - WebSocket / SSE
  - gRPC
  - Capsule protocol
    - CONNECT(-FOO)
  - etc
- Multi-stream context
  - WebTransport
    - MoQ (Media over QUIC)
    - etc

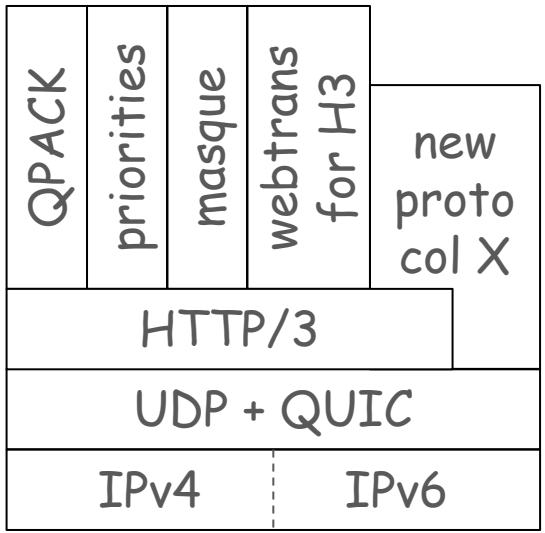
HTTP/2 streams

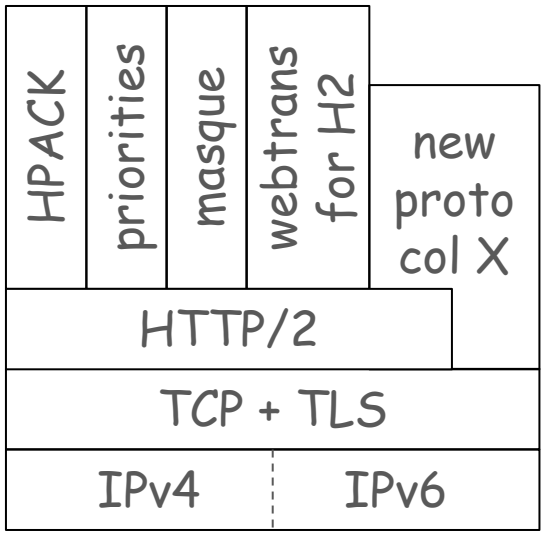
HTTP/3 streams



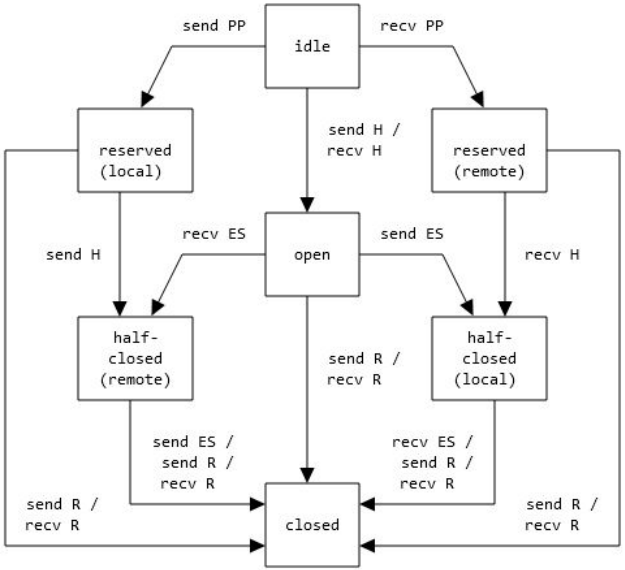
Sad state of application protocols

We now have to develop and maintain two different sets of stacks.

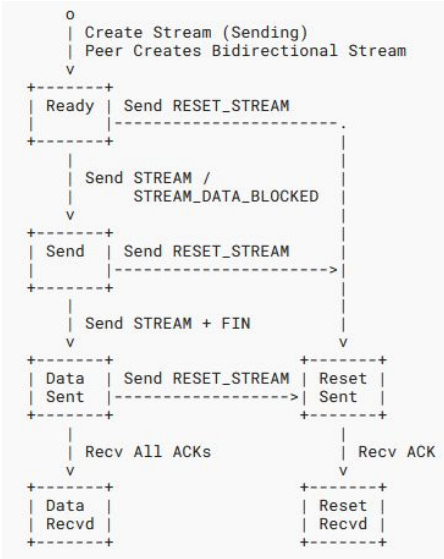


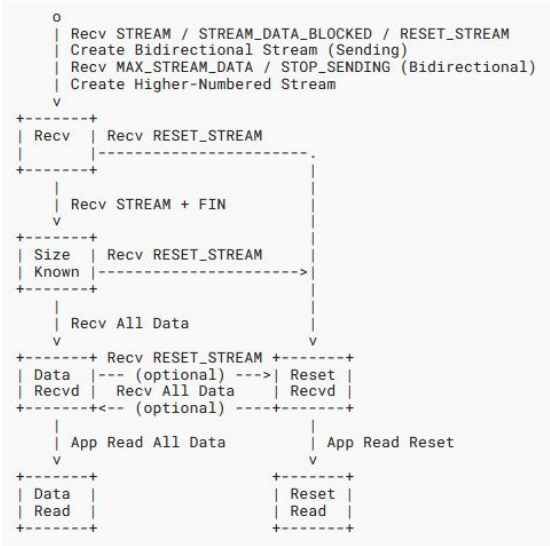


HTTP/2 streams



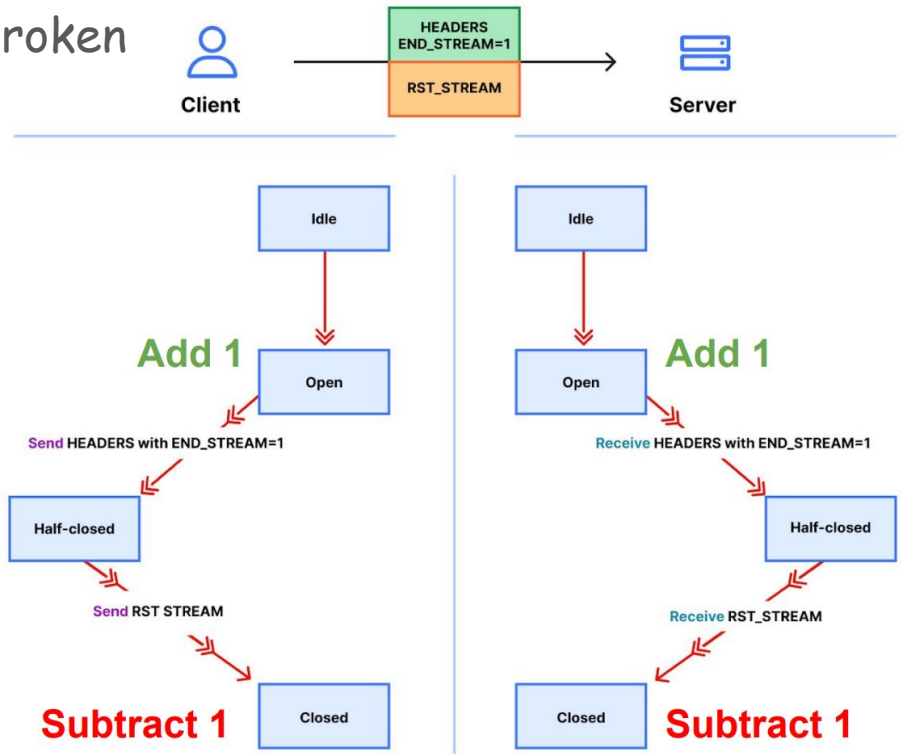
QUIC streams





HTTP/2 concurrency & rapid reset

- HTTP/2 stream concurrency is broken
- Default: "unlimited"
- De-facto client default: 100
- Servers picking < 100 break behaving clients
- Despite concurrency limits, misbehaving clients can create & reset streams at will



HTTP/3 concurrency & rapid reset

- HTTP/3 design is immune
- QUIC streams negotiate initial concurrency limit in handshake
- New stream credits must be issued before more streams can be created

Fix HTTP/2 concurrency

- Proposal to port QUIC stream concurrency to HTTP/2 as an extension: draft-thomson-httpbis-h2-stream-limits
  - Not much appetite

HTTP/2.1?

Maybe more appetite to take on bigger changes that add more value?
https://github.com/httpwg/admin/issues/56

1. Better Stream concurrency control
2. Remove server push
3. Remove RFC 7540 priorities (recommend RFC 9218?)
4. Restrict SETTINGS to once per connection at the start
5. Mandate extended CONNECT
6. Use QUIC variable-length integers (expand from 32-bit to 64-bit integers)
7. Grease every extension code point
8. Remove frame flags field - replace this with frame type instead (similar to QUIC and HTTP/3)
9. Remove CONTINUATION frames (larger frame sizes obviate them)
10. Mandate TLS; remove cleartext and upgrade (and hence avoid complications related to cleartext)
11. Remove prior knowledge and the preface stuff. (ALPN is enough)
12. Tweak so that stream errors allow eliciting an error HTTP response rather than just a RST_STREAM

Anything that isn't strictly HTTP semantics has to be **done twice** - once for HTTP/2 another for HTTP/3

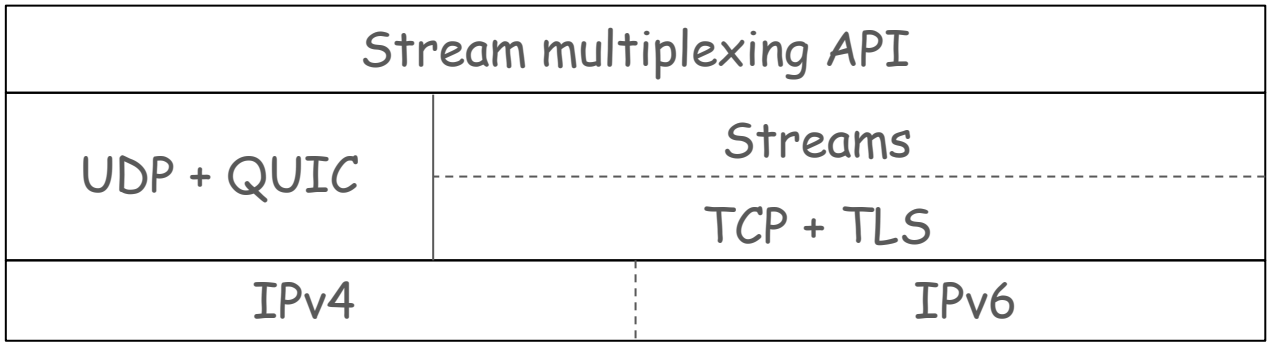
Everything we add, an intermediary needs to be able to translate, e.g.,

- QUIC DATAGRAM frames to capsules on HTTP streams
- WebTransport over H2 to WebTransport over H3
- RFC 7540 priorities to RFC 9218 priorities

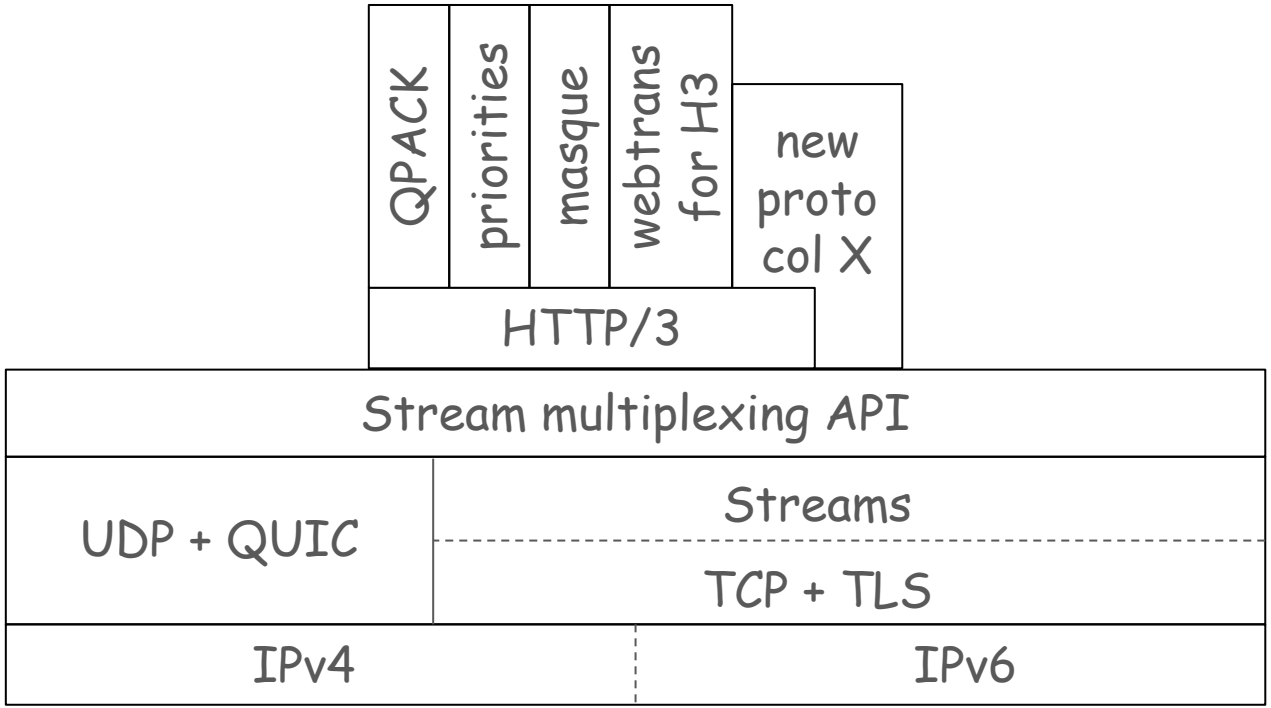
Multiplexing for 2025 and beyond

- What if we could take QUIC's streaming model and make it available to any form of transport?

Multiplexing for 2025 and beyond



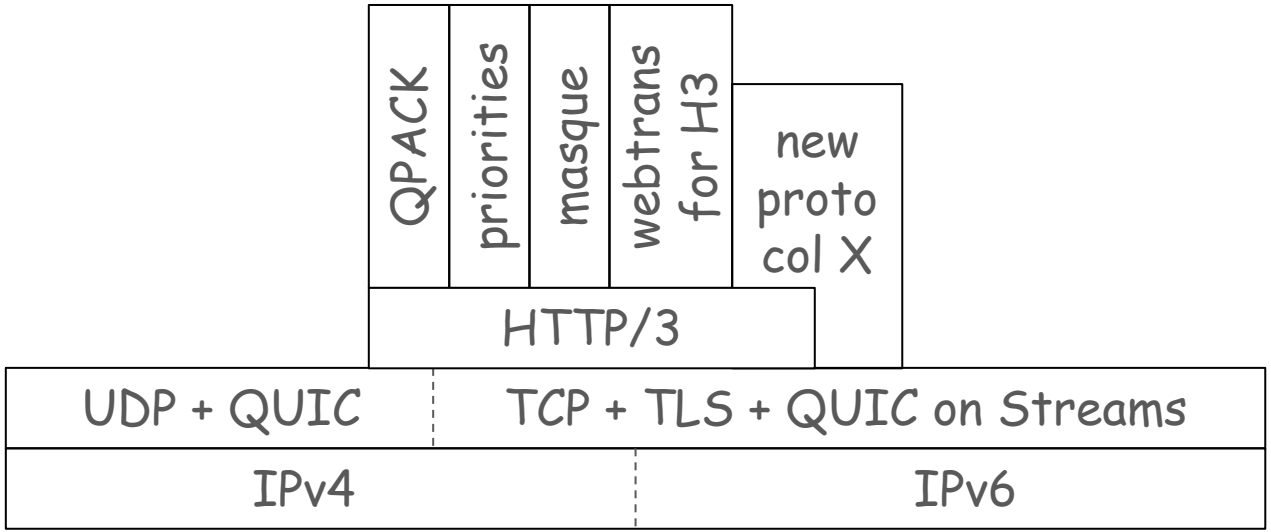
BYO protocol that needs multiplexing



QUIC on Streams

draft-kazuho-quic-quic-on-streams

draft-kazuho-httpbis-http3-on-streams



Our goals and non-goals

- Goals
  - eliminate the need to develop new things on top of two protocols
    - run unmodified HTTP/3 on top of QUIC on Streams
  - eliminate the need to deploy two HTTP versions
    - when you control both sides, this is immediate
  - reuse existing QUIC and HTTP/3 implementations
  - expose underlying transport properties for applications
- Non-goals
  - do not spend time optimizing TCP (e.g., solve HoL blocking) or QUIC frames. QUIC over UDP works in most cases and performs better, so QUIC on Streams is a fallback

Design of draft -00

- **new ALPN:** <insert bikeshed here>
- send **minimal set of QUICv1 frames** on top of TCP / TLS
  - only stream, datagram, and associated operations (flow control)
  - no ACK frames, CIDs, etc.
- Transport Parameters are exchanged using 1st frame called QS_TRANSPORT_PARAMETERS
- minimum of maximum frame size is 16KB (matches max. TLS record size)

※Working PoC for quicly created in 1/2 day. Took another 1/2 day to integrate that into H2O to run H3 client / server over QUIC on Streams.

※ Working PoC for quic-go

Concerns

- Are application layer protocols going to suffer if they're assuming UDP and get TCP?
  - Performance versus HTTP/2
- Yet another way to do HTTP, one more protocol variant
  - Introducing a whole new set of [security] problems
- Possibility that networks will no longer feel the pressure to make QUIC/UDP work


Pokémon
SHINING
PEARL