

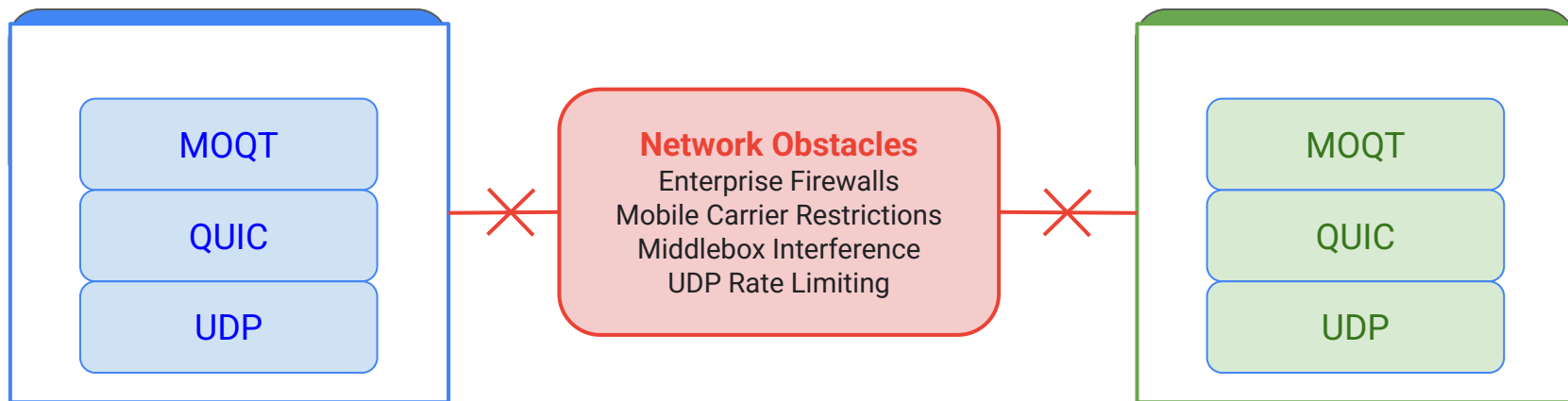
MOQT over QMux Transport

Enabling MOQT over Reliable Byte Streams

IETF 125

Authors: Suhas Nandakumar, Cullen Jennings

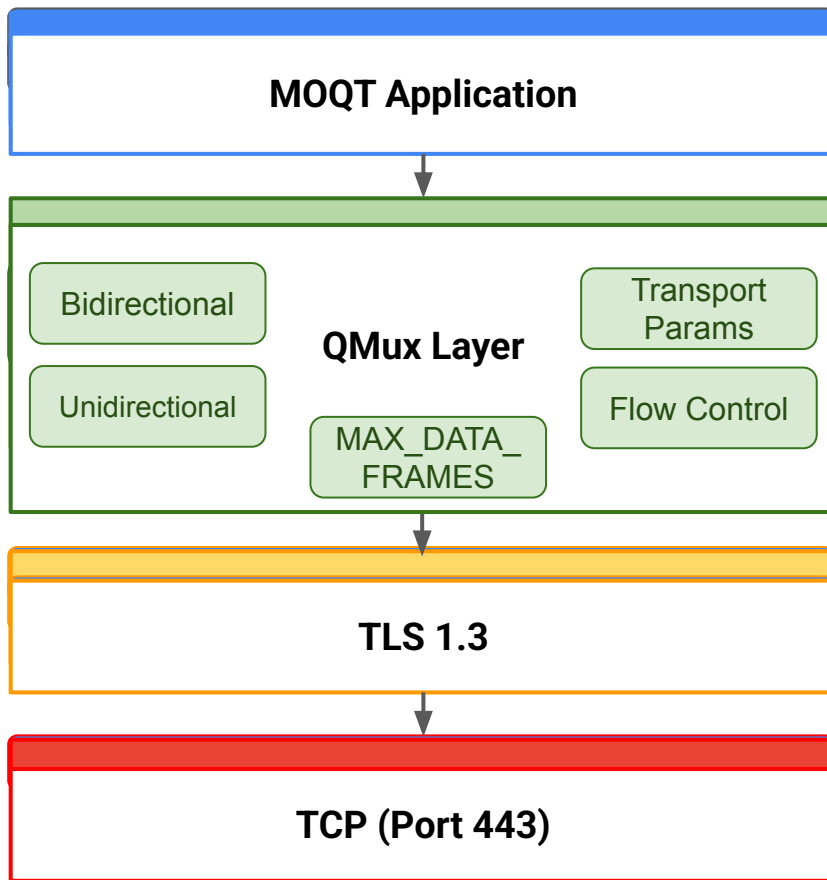
The Problem: UDP Connectivity Challenge



Challenge: MOQT designed for QUIC (UDP), but many networks often block QUIC and/or UDP

The Solution: QMux Transport Layer

Key Insight
Applications coded
against QUIC APIs work
unchanged



Why QMux for MOQT?

Fallback

Many cases where UDP blocked but TCP port 443 works

Single Codebase

Same MOQT logic for QUIC and TCP.

Infrastructure Reuse

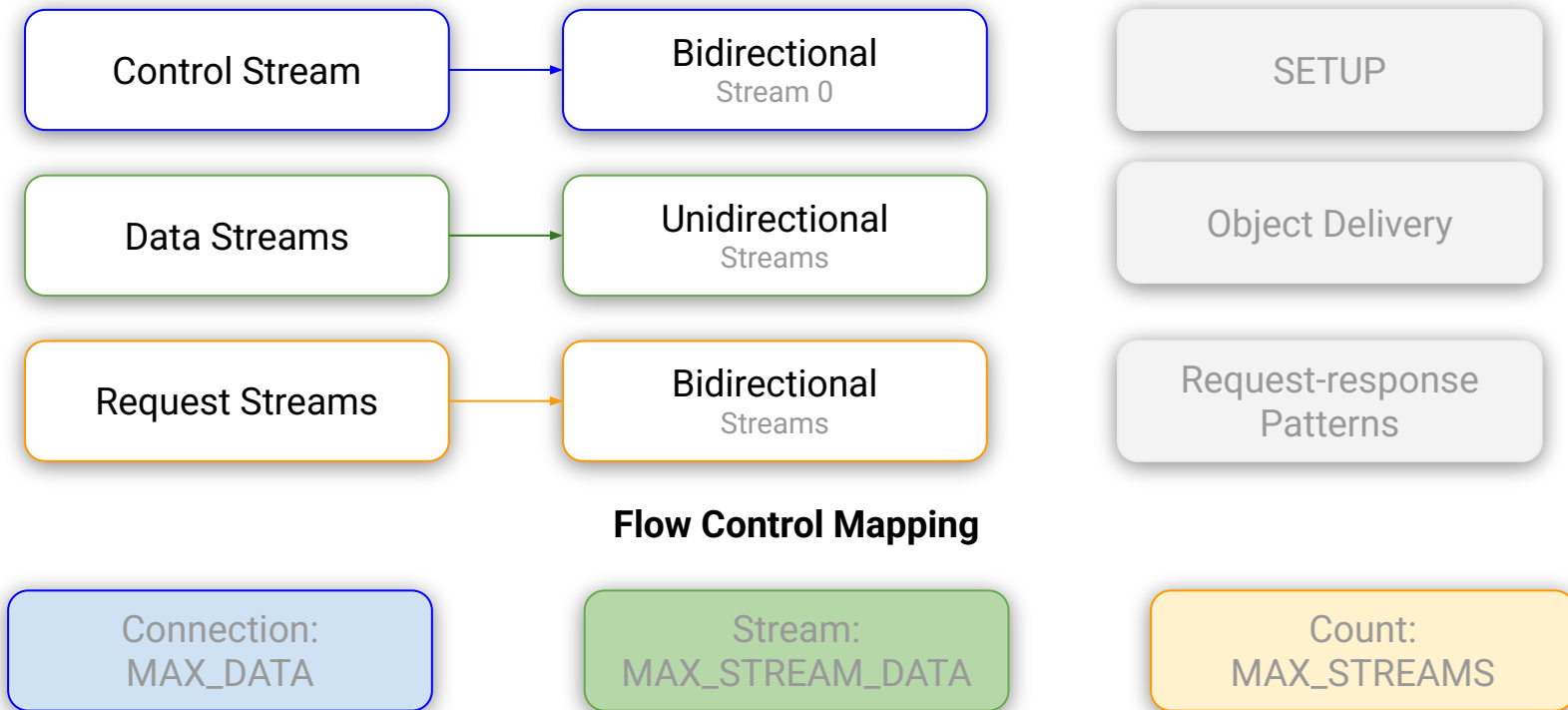
Existing load balancers, CDNs, TLS accelerators, and proxies

Graceful Degradation

Better than no connection.
Seamless failover from QUIC

- ! Head-of-line blocking (TCP packet loss blocks all streams)
- ! No connection migration
- ! 1-2 RTT additional latency on initial connection

Stream Mapping: MOQT to QMUX



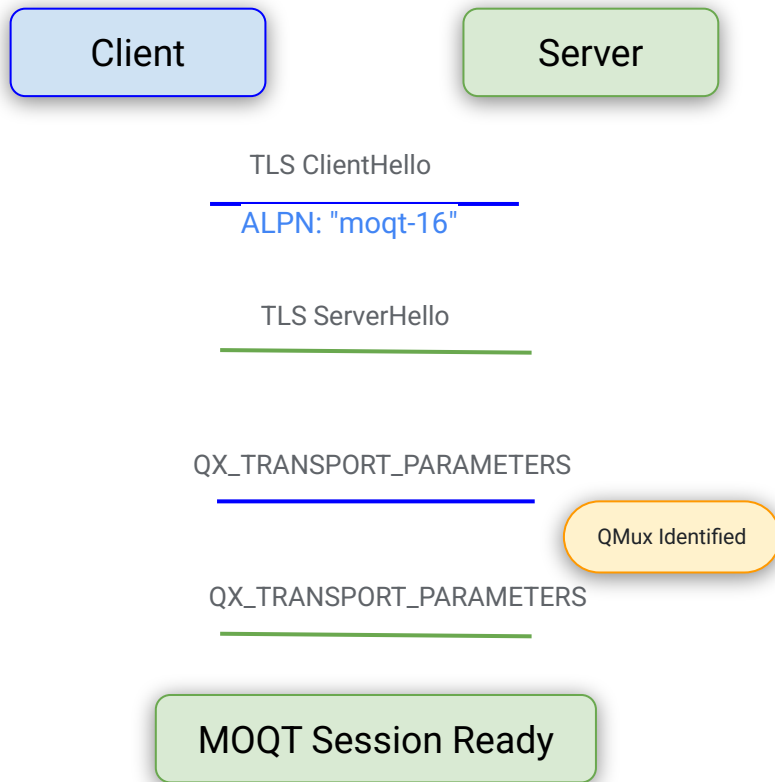
ALPN: Current Draft Proposal

Same ALPN for all Transports

Type	ALPN Value
Final	moqt
Drafts	moqt-XX (e.g., moqt-16)

Key Points

- Transport distinguished by connection type (QUIC vs TCP+TLS)
- QMux identified via QX_TRANSPORT_PARAMETERS exchange
- Simplified client implementation
- Enables seamless fallback



ALPN: Alternative Proposals

Challenge: Stack Identification

Stack	Layers
A	MOQT → QUIC
B	MOQT → QMux → TCP/TLS
C	MOQT → WebTransport → QUIC
D	MOQT → WebTransport → QMux → TCP/TLS

Protocol/Version Negotiation #1

Design inspired by “WT-Available-Protocols” for WebTransport

MOQT over QMux

TLS ALPN in TLS: “qmux-1”

Next-Available-Protocols in QX_Paramertes: “moqt-16, moqt-17”

MOQT over WebTransport ovr QMux

TLS ALPN in TLS: “qmux-1”

Next-Available-Protocols in QX_Parameters: “h3”

WT-Available-Protocols in Web transport : “moqt-16, moqt-17”

Protocol/Version Negotiation #2

Compound ALPN (moqt-xx-qmux-yy)

MOQT over WebTransport over QMux

TLS ALPN: "h3"

WT-Available-Protocols: "moqt-16-qmux-1,
moqt-17-qmux-2"

MOQT over QMux (native)

TLS ALPN: "moqt-16-qmux-1", "moqt-17-qmux-2"

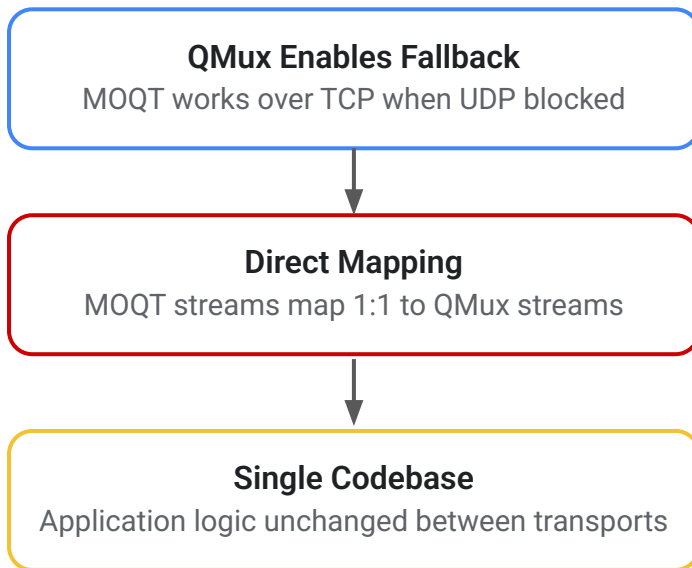
Challenge: Version Explosion

MOQT	QMux	Combinations
16, 17	1	2 ALPNs
16, 17	1, 2	4 ALPNs
16, 17, 18	1, 2	6 ALPNs!

PRO: All version info at TLS/WT layer

CON: N × M ALPN values, coordination overhead

Summary



Open Questions for Discussion

- Should QMux use same ALPN as native QUIC (moqt-XX)?
- How should clients discover server QMux support?
- WebTransport over QMux: use WT-Available-Protocols?
- Any concerns with early version negotiation approach?

Next Steps

- Is there interest in pursuing this work in the WG ?
 - Early deployments will do something, risk is what they do is not easy to transition to IETF design
- Inviting co-authors and reviewers interested in pursuing this work ?