The New Frontier: Protocol Mappings

Tommy Pauly
TAPS
IETF 111, July 2021, Virtual
Where are we in TAPS?

- 2017: RFC 8095: Survey of IETF services
- 2018: RFC 8303: Transport features, RFC 8304: Features (UDP)
- 2020: RFC 8922: Security + Transport
- 2021+: Architecture, Interface, Implementation

Mappings?
What is a mapping to TAPS?

Defines how a transport protocol interprets API calls

Defines special interactions with properties

Allows specification of protocol-specific properties
Appendix A. API Mapping Template

Any protocol mapping for the Transport Services API should follow a common template.

Connectedness: (Connectionless/Connected/Multiplexing Connected)

Data Unit: (Byte-stream/Datagram/Message)

Connection Object:

Initiate:

InitiateWithSend:

Ready:

InitiateError:

ConnectionError:

Listen:

ConnectionReceived:

Clone:

Send:

Receive:

Close:

Abort:
Existing mappings

Implementation draft defines base mappings

TCP

MPTCP

UDP / UDP-Lite

UDP Multicast Receive

SCTP
Future work

10 open issues tagged for “mappings”

QUIC

HTTP/3 and HTTP/2 streams

HTTP/1.1 pipelining

WebRTC

Advanced TLS features (ticket requests, etc)

WebTransport is also a good candidate
Case study: QUIC

Old draft from 2018 (draft-pauly-quic-interface-00)

QUIC and TAPS have both evolved since

Network.framework now supports QUIC, by mapping a QUIC stream to a TAPS Connection
Case study: QUIC

Initiate() -> Allocate stream, QUIC handshake if needed

Clone() -> Allocate stream on existing connection

Send() -> STREAM frame, complete sends FIN

Receive() -> Handle STREAM frames

Close() -> RESET_STREAM

CloseGroup() -> CONNECTION_CLOSE

...
WG questions

Should we add mappings to our charter?

Which mappings belong in TAPS, versus other groups?

Should the mappings have a registry?