An Unreliable Datagram Extension to QUIC

draft-pauly-quic-datagram

Tommy Pauly (tpauly@apple.com)
Eric Kinnear (ekinnear@apple.com)
David Schinazi (dschinazi.ietf@gmail.com)

QUIC

IETF 106, November 2019, Singapore

Motivation

Unreliable data transmission supports many use cases

- Applications that need a reliable control stream and unreliable flows
- Media streaming, gaming, VPN-style tunneling, and more

QUIC provides functionality beyond that of DTLS, UDP

Let’s use the QUIC extension mechanism!

Why Datagrams in QUIC?

Share a single handshake and authentication context for reliable stream data and unreliable datagram data

QUIC handshake has more nuanced loss recovery during the handshake compared to DTLS

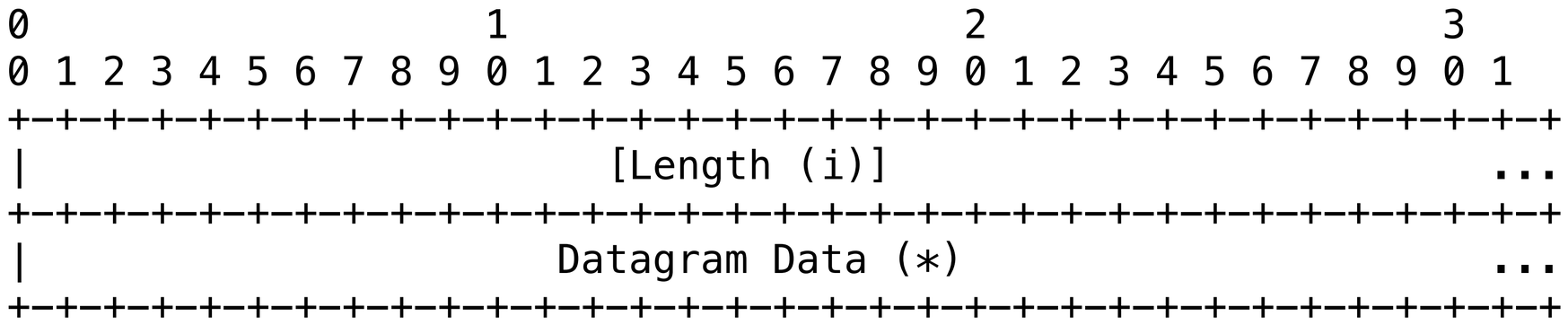
Use QUIC features not present in alternatives

- Transport parameters
- Transport level acknowledgements of datagram data
- Multiplexing of additional content over same transport

Design

DATAGRAM frame (0x30 and 0x31)

Length field is optional, determined by least significant bit



Negotiated via max_datagram_frame_size transport parameter

Design Details

`DATAGRAM` frames are ack-eliciting and not retransmitted

- Just like `PING`

`DATAGRAM` frames do not contribute to flow control limits

Flow IDs are gone

- Didn't go far, see draft-schinazi-quic-h3-datagram-02

`max_datagram_frame_size` can be stored for 0-RTT

Implementation Status

Supported by multiple implementations

quiche (SiDUCK), aioquic, Google QUIC, AppleQUIC

Wireshark can dissect `DATAGRAM` frames

Achieved interop between quiche and aioquic during the hackathon

Questions? Adoption?

DATAGRAM Frame

(0x30 and 0x31)