Optimizing Proxies for QUIC

Tommy Pauly
MASQUE
IETF 108, July 2020, Virtual
Optimization Opportunity

CONNECT-UDP allows tunneling UDP in H3

Adds extra encapsulation and encryption for proxied QUIC flows

Traditional TCP proxies do not require this overhead
Single-Hop Proxying

Optimize for a target that supports QUIC
Multi-Hop Proxying

Optimize for multiple MASQUE proxies
CONNECT-QUIC

QUIC packets are either

1. Tunneled: client ↔ target Long Header packets encapsulated in client ↔ proxy DATAGRAMs

2. Forwarded: client ↔ target Short Header packets sent directly on client ↔ proxy UDP socket
CONNECT-QUIC Requests

Map one direction of a Connection ID to a proxied flow, with a DATAGRAM ID to use for Long Header packets.

When the request STREAM closes, remove mapping

Client-Connection-Id = sf-binary

Server-Connection-Id = sf-binary

Datagram-Flow-Id = sf-integer
Request format

Proxy Client CID 0x31323334 to target.example.com:443, use DATAGRAM flow 1:

HEADERS + END_HEADERS
:method = CONNECT-QUIC
:authority = target.example.com:443
client-connection-id = :MTIzNA==:
datagram-flow-id = 1
MASQUE Setup
TCP-through-MASQUE

Origin using TLS/TCP
2 RTT for data
1 RTT with resumption
QUIC-through-MASQUE

Origin using QUIC
2 RTT for data
1 RTT with resumption
Merging with CONNECT-UDP

Very similar to the CONNECT-UDP case, which always tunnels

CONNECT-UDP should be extensible, and maybe generalized

CONNECT-QUIC needs to fall back to tunneling only (CONNECT-UDP) if not using H3
Open Issues

Allowing proxies to reuse next-hop UDP sockets

Mapping Connection ID trickiness

Interactions with load balancers

Avoid forwarding replayed or injected packets