QUIC-Aware Proxying Using HTTP

Eric Rosenberg, Tommy Pauly, and David Schinazi
MASQUE
IETF 115, November 2022
Why QUIC-Aware?

When transmitting UDP over CONNECT-UDP tunnels

• IPv4 port exhaustion between proxy and target

• MTU loss (~30-45 bytes) per proxy
  
  • With initial 1350 MTU, may not be able to exceed ~3 hops without violating QUIC’s 1200 requirement

• QUIC processing and UDP send/receive overhead
QUIC-Aware Proxying

- Client tells proxy about inner QUIC connection’s CIDs (using capsules!)
- Proxy may reuse target-facing ports
- Client and proxy may skip encapsulation and encryption for proxied SH packets — avoiding cumulative MTU overhead issues
- Forwarded mode packets on the wire use virtual CIDs instead of the inner connection’s real CIDs
QUIC-Aware Proxying

Applicability of Forwarded Mode

• Mostly useful in multi-hop environments — particularly those where cumulative MTU loss makes a difference

• QUIC packet contents identical on the client<->proxy and proxy<->target paths. Does not prevent traffic analysis by observers of both sides of the proxy — although, timing analysis still possible even without forwarded mode.
QUIC-Aware Proxying

Client

CONNECT UDP + CID Capsules

LH Packets

SH Packets (Virtual CID)

Proxy

HTTP Stream

Forwarded

Target

LH Packets

SH Packets (Real CID)

HTTP Datagrams
QUIC-Aware Proxying

Virtual Connection IDs

- Compatibility with load balancers
- CID bytes change on connection migration
- Avoid trivial linkability via CID, although content still linkable
Performance in Lab

- Quiche-based client, proxy, and origin
- Linux XDP\(^1\) hook with eBPF\(^2\) program to route packets based on CID
- Single 100GbE link

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
<th>Gbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tunneled</td>
<td>90%</td>
<td>52</td>
</tr>
<tr>
<td>Forwarded</td>
<td>1%</td>
<td>91</td>
</tr>
</tbody>
</table>

---


draft-pauly-masque-quic-proxy - IETF 115
# Connection ID Exchange

<table>
<thead>
<tr>
<th>Capsule Type</th>
<th>Sender</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>REGISTER_TARGET_CID</td>
<td>Client</td>
<td>Target CID and Stateless Reset Token</td>
</tr>
<tr>
<td>REGISTER_CLIENT_CID</td>
<td>Client</td>
<td>Client CID, Virtual Client CID, and Stateless Reset Token</td>
</tr>
<tr>
<td>ACK_TARGET_CID</td>
<td>Proxy</td>
<td>Target CID, Virtual Target CID, and Stateless Reset Token</td>
</tr>
<tr>
<td>ACK_CLIENT_CID</td>
<td>Proxy</td>
<td>Client CID</td>
</tr>
<tr>
<td>CLOSE_TARGET_CID</td>
<td>Either</td>
<td>Target CID</td>
</tr>
<tr>
<td>CLOSE_CLIENT_CID</td>
<td>Either</td>
<td>Client CID</td>
</tr>
</tbody>
</table>
CONNECT-UDP + Proxy-QUIC-Forwarding=1 + Capsule \{ REGISTER_CLIENT_CID \} + HTTP Datagram \{ QUIC Initial \}

**Construct Socket, Port Reuse, Target->Client Forwarding Rule**

QUIC Initial

200 OK + Proxy-QUIC-Forwarding=1 + Capsule \{ ACK_CLIENT_CID \}

HTTP Datagram \{ QUIC Initial \}

Capsule \{ REGISTER_TARGET_CID \}

**Client->Target Forwarding Rule**

Capsule \{ ACK_TARGET_CID \}

QUIC SH with Virtual Target CID

QUIC SH with Real Target CID

QUIC SH with Virtual Client CID

QUIC SH with Real Client CID
Status

• Looking for review and feedback

• MASQUE working group adoption?