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

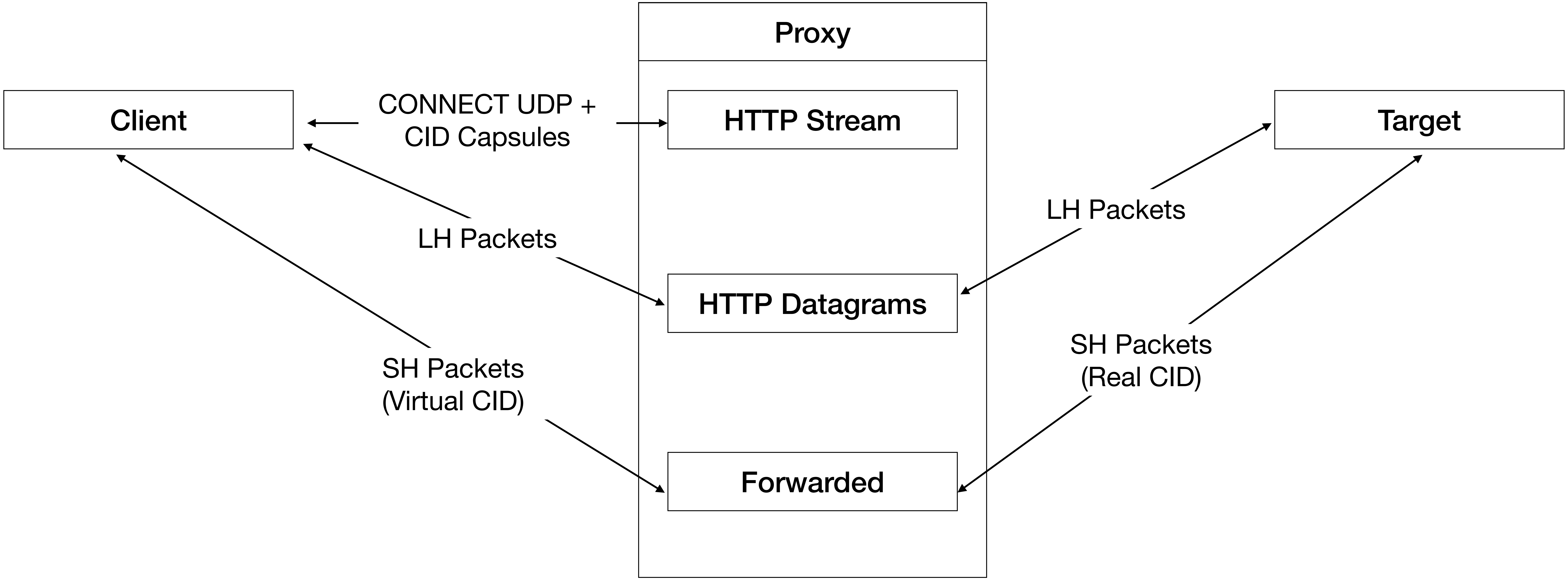
QUIC-Aware Proxying

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



Proxy

Client

CONNECT UDP +
CID Capsules

HTTP Stream

Target

LH Packets

LH Packets

HTTP Datagrams

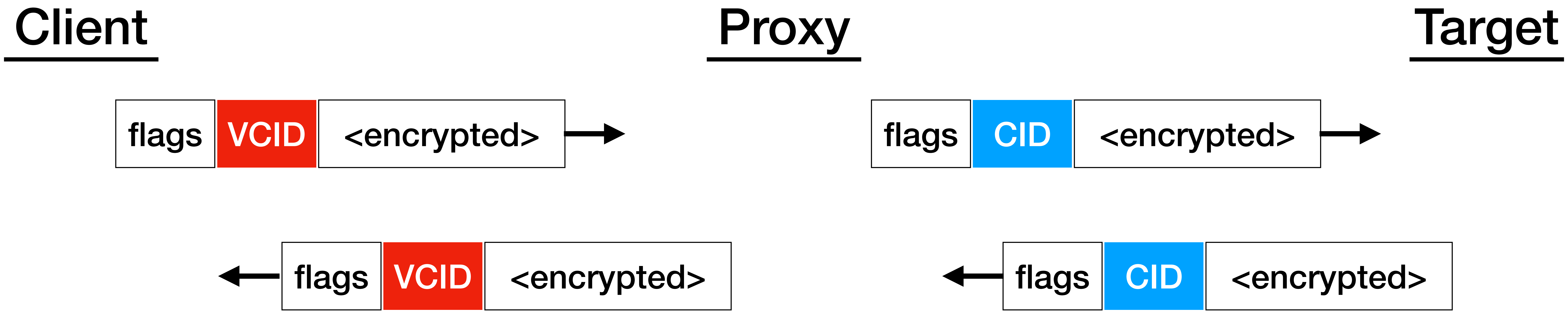
SH Packets
(Real CID)

SH Packets
(Virtual CID)

Forwarded

QUIC-Aware Proxying

Virtual Connection IDs



Client

Proxy

Target

flags VCID <encrypted>

flags CID <encrypted>

flags VCID <encrypted>

flags CID <encrypted>

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

Performance in Lab

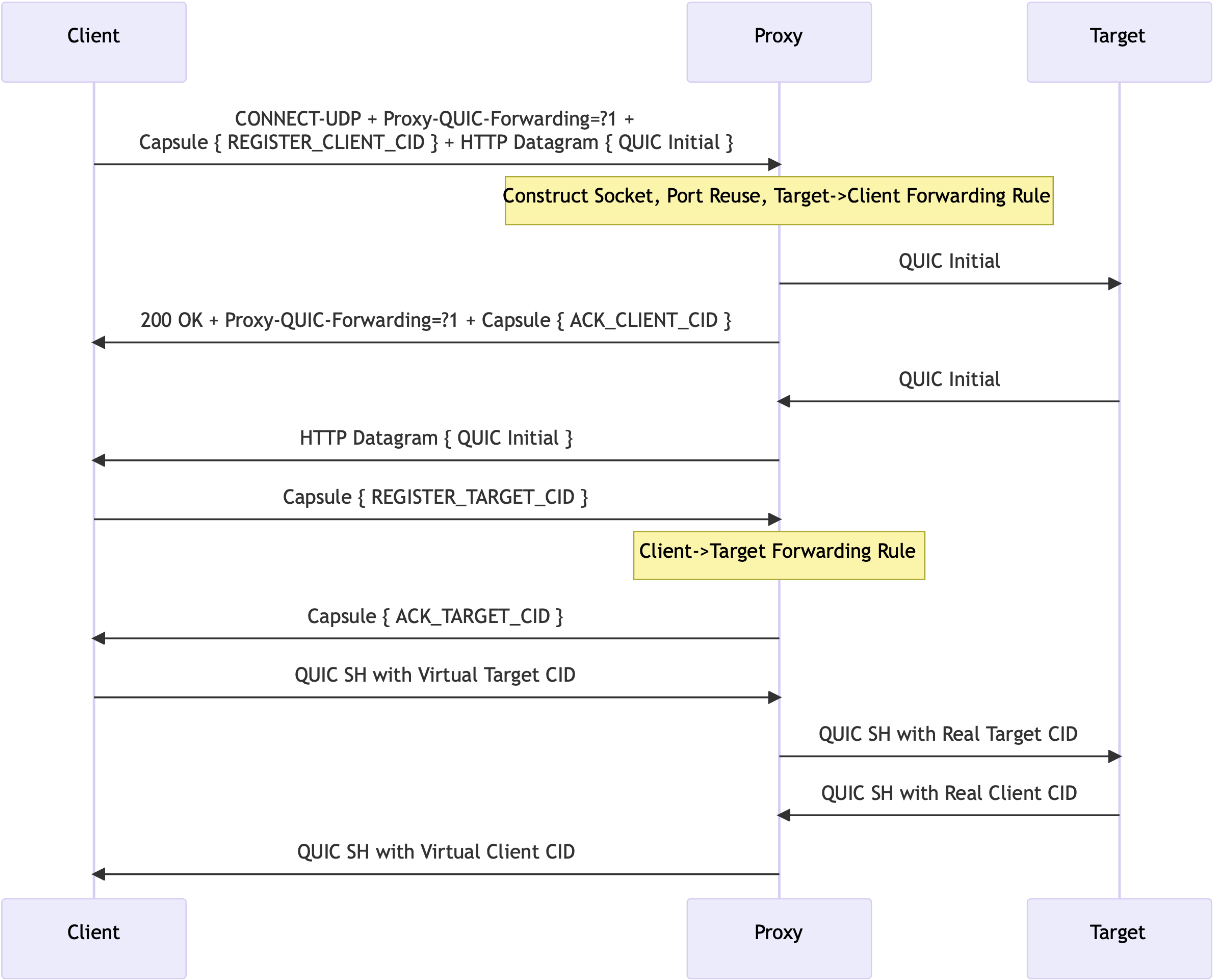
- Quiche-based client, proxy, and origin
- Linux XDP[1] hook with eBPF[2] program to route packets based on CID
- Single 100GbE link

	CPU	Gbps
Tunneled	90%	52
Forwarded	1%	91

[1] eXpress Data Path, https://www.iovisor.org/technology/xdp
[2] Extended Berkeley Packet Filter, https://www.kernel.org/doc/html/latest/bpf/index.html

Connection ID Exchange

Capsule Type	Sender	Contents
REGISTER_TARGET_CID	Client	Target CID and Stateless Reset Token
REGISTER_CLIENT_CID	Client	Client CID, Virtual Client CID, and Stateless Reset Token
ACK_TARGET_CID	Proxy	Target CID, Virtual Target CID, and Stateless Reset Token
ACK_CLIENT_CID	Proxy	Client CID
CLOSE_TARGET_CID	Either	Target CID
CLOSE_CLIENT_CID	Either	Client CID



Client

Proxy

Target

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 }

QUIC Initial

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 Real Client CID

QUIC SH with Virtual Client CID

Client

Proxy

Target

Status

- Looking for review and feedback
- MASQUE working group adoption?