The CONNECT-IP method for proxying IP traffic

Mirja Kühlewind
Magnus Westerlund
Marcus Ihlar
Zaheduzzaman Sarker

draft-kuehlewind-masque-connect-ip-00

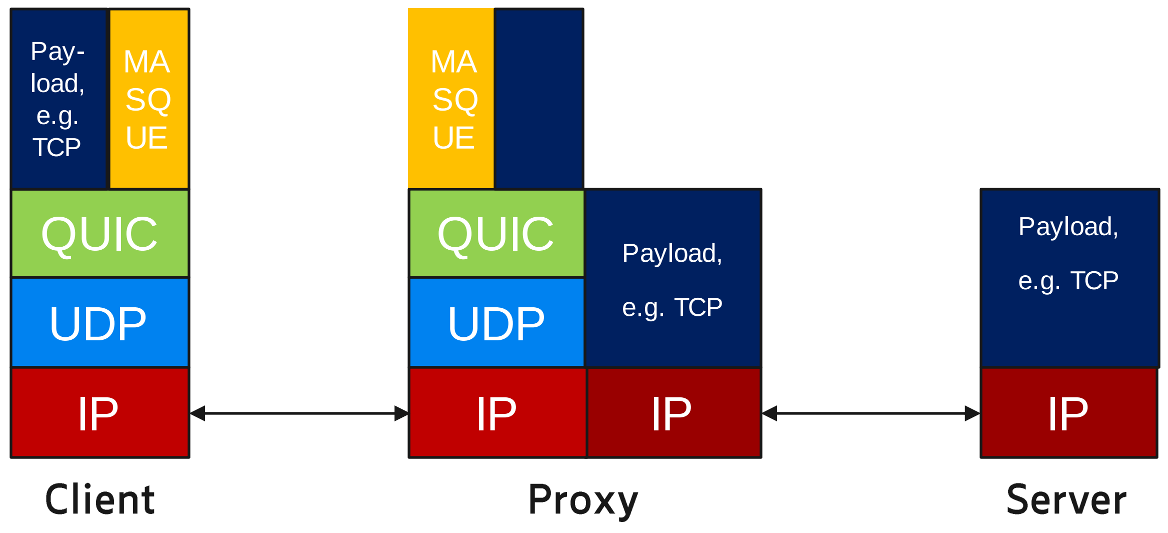
Outline

- Approach: IP payload forwarding
- Proposed MASQUE CONNECT-IP method
- Review of requirements on IP Proxying
- Additional Considerations (might apply also for CONNECT-UDP)

This presentation aims to create discussion and more in-depth understanding about different approaches to IP proxying
1) as input for requirements on IP proxying and
2) to understand similarities/differences to UDP proxying

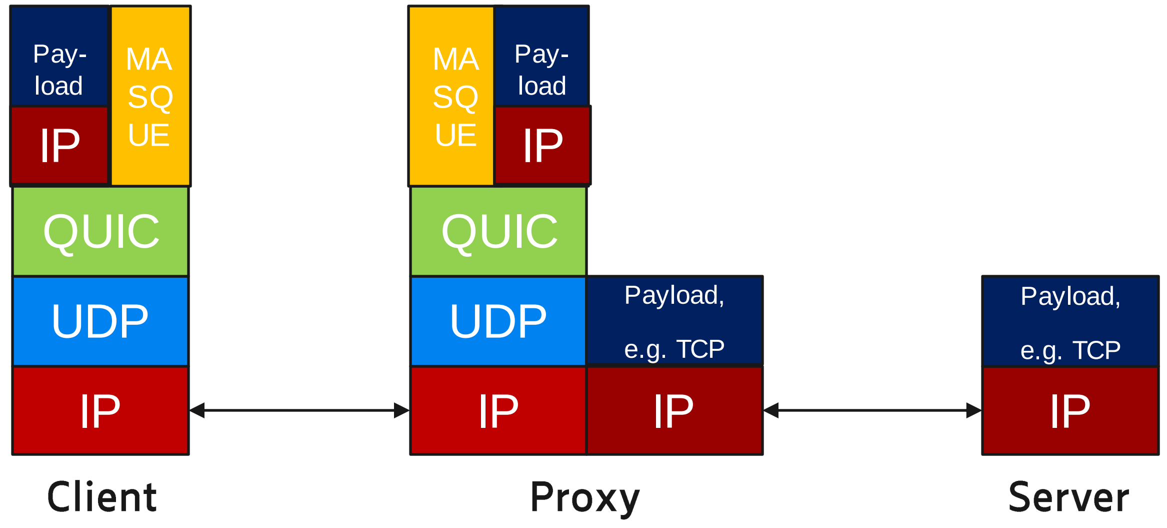
Two possible design approaches

In draft: IP payload forwarding



- Client only provides target IP address (and other relevant information) with CONNECT-IP request
  - Goal: reduce packet overhead
  - Note: Reuse of functions needed for CONNECT-UDP
- Proxy constructs and adds IP header/selects src IP address
- Stateless forwarding of incoming traffic not considered (might be needed for network-to-network use case)

Alternative: IP packet (incl. header) forwarding



- IP header is part of the QUIC tunnel payload
- Easier for Network-to-Network: client provides IP range
- Need for source address validation (or NAT)
- Additonal signalling for route negotiation for prefixes

Brief overview: CONNECT-IP method for IP Payload forwarding

```
CONNECT-IP target.example.com
IP-Protocol: 6
```

- Support of stream-based and datagram-based modes
  - Use of HTTP DATA frames to indicate IP payload length in stream mode
  - `Datagram-Flow-Id` Header for datagram mode (as in CONNECT-UDP)
- Optional `Conn-ID` Header to provide additional flow identifier field in payload, e.g. port number for TCP or UDP (similar to QUIC-aware proxying but generic)
- MASQUE signalling (`HTTP POST` or `GET`)
  - Initial negotiation and capability notification: `/.well-known/masque/config`
  - Per-forwarding association (based on events, e.g. ECN, ICMP): `/.well-known/masque/<id>`

Requirements on IP Proxying from draft-ietf-masque-ip-proxy-reqs-00

- **Proxying of IP packets:** "The protocol will establish Data Transports, which will be able to forward IP packets, in their unmodified entirety."
  - ➢ IP payload forwarding can reuse CONNECT-UDP functions and reduces overhead
- **IP Assignment:** "The client will be able to request to be assigned an IP address range, optionally specifying a preferred range." "Similarly, to support the network-to-network use case, the server will be able to request assignment of an IP address range"
  - ➢ Proxy performs IP address selection e.g. to route return traffic to proxy in Consumer VPN use case
- **Route Negotiation:** "At any point in an IP Session (not limited to its initial negotiation), the protocol will allow both client and server to inform its peer that it can route a set of IP prefixes."
  - ➢ GET/POST could be used to exchange configuration files but also requires return path validation…?
- **Load balancing:** "Clients and servers should each be able to instantiate new Data Transports."
  - ➢ No support for stateless forwarding of incoming traffic; additional proxy-client signaling required
- **Non-requirement - Translation:** "Some servers may wish to perform Network Address Translation (NAT) or any other modification to packets they forward. Doing so is out of scope for the proxying protocol."
  - ➢ This is required by Consumer VPN use case (e.g. if address is obfuscated) and should be exposed to client (which might also require a stable address?)

Requirements and open issues that also apply for CONNECT-UDP

- **Maximum Transmission Unit:** "The protocol will allow endpoints to inform each other of the Maximum Transmission Unit (MTU) they are willing to forward."
  - ➢GET/POST to exchange configuration files
  - ➢also issue #7 CONNECT-UDP draft
- **Extensibility: "**Once the session is established, the protocol will provide a mechanism that allows reliably exchanging vendor-specific messages in both directions at any point in the lifetime of the IP Session."
  - ➢GET/POST to exchange per-forwarding flow configuration files
  - ➢Alternatively: use new HTTP control frames to be interleaved with data on forwarding stream

Additional Considerations (also apply for CONNECT-UDP)

HTTP Consideration

- Use of HTTP DATA frames on streams is inline with HTTP/2 & HTTP/3 CONNECT -> supports extensibility (issue #15)
- Handling of HTTP datagram flow ID: client-based ID selection and handling of conflicts (see also draft-pauly-masque-quic-proxy)

QUIC handling

- Is parallel use of stream and datagram modes possible?
- Forwarding state lifetime management is bounded to stream (issue #4 CONNECT-UDP draft)
- Server-facing socket handling, and DNS lookups…

MASQUE Signalling

- Event-based vs per-packet signalling? Proposed ECN handling based on event notifications. Path validation?
- Which ICMP information are consumed by the client or which ones need to be handled by the proxy?
- Use of a dedicated signalling control stream vs. interleaving of control message on the forwarding stream?