Workgroup: MASQUE
Internet-Draft:
draft-pauly-masque-quic-proxy-01
Published: 13 April 2021
Intended Status: Experimental
Expires: 15 October 2021
Authors: T. Pauly   D. Schinazi
         Apple Inc.   Google LLC

QUIC-Aware Proxying Using CONNECT-UDP

Abstract

This document defines an extension to the CONNECT-UDP HTTP method that adds specific optimizations for QUIC connections that are proxied. This extension allows a proxy to reuse UDP 4-tuples for multiple connections. It also defines a mode of proxying in which QUIC short header packets can be forwarded through the proxy rather than being re-encapsulated and re-encrypted.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/tfpauly/quic-proxy.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The CONNECT-UDP HTTP method [CONNECT-UDP] defines a way to send datagrams through an HTTP proxy, where UDP is used to communicate between the proxy and a target server. This can be used to proxy QUIC connections [QUIC], since QUIC runs over UDP datagrams.

This document uses the term "target" to refer to the server that a client is accessing via a proxy. This target may be an origin hosting content, or another proxy.

This document extends the CONNECT-UDP HTTP method to add signalling about QUIC Connection IDs. QUIC Connection IDs are used to identify QUIC connections in scenarios where there is not a strict bidirectional mapping between one QUIC connection and one UDP 4-tuple (pairs of IP addresses and ports). A proxy that is aware of Connection IDs can reuse UDP 4-tuples between itself and a target for multiple proxied QUIC connections.

This extension is only defined for HTTP/3 [HTTP3] and not any earlier versions.

Awareness of Connection IDs also allows a proxy to avoid re-encapsulation and re-encryption of proxied QUIC packets once a connection has been established. When this functionality is present, the proxy can support two modes for handling QUIC packets:

1. Tunnelled, in which client <-> target QUIC packets are encapsulated inside client <-> proxy QUIC packets. These packets use multiple layers of encryption and congestion control. QUIC long header packets MUST use this mode. QUIC short header packets MAY use this mode. This is the default mode for CONNECT-UDP.

2. Forwarded, in which client <-> target QUIC packets are sent directly over the client <-> proxy UDP socket. These packets are only encrypted using the client-target keys, and use the client-target congestion control. This mode MUST only be used for QUIC short header packets.

Forwarding is defined as an optimization to reduce CPU processing on clients and proxies, as well as avoiding MTU overhead for packets on the wire. This makes it suitable for deployment situations that otherwise relied on cleartext TCP proxies, which cannot support QUIC and have inferior security and privacy properties.

The properties provided by the forwarding mode are as follows:

*All packets sent between the client and the target traverse through the proxy device.

*The target server cannot know the IP address of the client solely based on the proxied packets the target receives.

*Observers of either or both of the client <-> proxy link and the proxy <-> target are not able to learn more about the client <-> target communication than if no proxy was used.

It is not a goal of forwarding mode to prevent correlation between client <-> proxy and the proxy <-> target packets from an entity that can observe both links. See Section 9 for further discussion.

Both clients and proxies can unilaterally choose to disable forwarded mode for any client <-> target connection.

QUIC proxies only need to understand the Header Form bit, and the connection ID fields from packets in client <-> target QUIC connections. Since these fields are all in the QUIC invariants header [INVARIANTS], QUIC proxies can proxy all versions of QUIC.

1.1. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

This document uses the following terms:

- Client: the client of all QUIC connections discussed in this document.

- Proxy: the endpoint that responds to the CONNECT-UDP request.

- Target: the server that a client is accessing via a proxy.

- Client <-> Proxy QUIC connection: a single QUIC connection established from the client to the proxy.

- Datagram flow ID: represents a flow of HTTP/3 DATAGRAMs [H3DGRAM] specific to a single client <-> proxy QUIC connection.

- Socket: a UDP 4-tuple (local IP address, local UDP port, remote IP address, remote UDP port). In some implementations, this is referred to as a "connected" socket.

- Client-facing socket: the socket used to communicate between the client and the proxy.

- Server-facing socket: the socket used to communicate between the proxy and the target.

- Client Connection ID: a QUIC Connection ID that is chosen by the client, and is used in the Destination Connection ID field of packets from the target to the client.

*Server Connection ID: a QUIC Connection ID that is chosen by the target, and is used in the Destination Connection ID field of packets from the client to the target.

2. Required Proxy State

In the methods defined in this document, the proxy is aware of the QUIC Connection IDs being used by proxied connections, along with the sockets used to communicate with the client and the target. Tracking Connection IDs in this way allows the proxy to reuse server-facing sockets for multiple connections and support the forwarding mode of proxying.

QUIC packets can be either tunnelled within an HTTP/3 proxy connection using QUIC DATAGRAM frames [DGRAM], or be forwarded directly alongside an HTTP/3 proxy connection on the same set of IP addresses and UDP ports. The use of forwarded mode requires the consent of both the client and the proxy.

In order to correctly route QUIC packets in both tunnelled and forwarded modes, the proxy needs to maintain mappings between several items. There are three required unidirectional mappings, described below.

2.1. Datagram Flow ID Mapping

Each pair of client <-> proxy QUIC connection and datagram flow ID MUST be mapped to a single server-facing socket.

```
(Client <-> Proxy QUIC connection + Datagram flow ID)
    => Server-facing socket
```

Multiple datagram flow IDs can map to the same server-facing socket, but a single datagram flow ID cannot be mapped to multiple server-facing sockets.

This mapping guarantees that any QUIC packet using the datagram flow ID sent from the client to the proxy in tunnelled mode can be sent to the correct target.

2.2. Server Connection ID Mapping

Each pair of Server Connection ID and client-facing socket MUST map to a single server-facing socket.

```
(Client-facing socket + Server Connection ID)
    => Server-facing socket
```

Multiple pairs of Connection IDs and sockets can map to the same server-facing socket.

This mapping guarantees that any QUIC packet containing the Server Connection ID sent from the client to the proxy in forwarded mode can be sent to the correct target. Thus, a proxy that does not allow forwarded mode does not need to maintain this mapping.

2.3. Client Connection ID Mappings

Each pair of Client Connection ID and server-facing socket MUST map to a single datagram flow ID on a single client <-> proxy QUIC connection. Additionally, the pair of Client Connection ID and server-facing socket MUST map to a single client-facing socket.

```
(Server-facing socket + Client Connection ID)
    => (Client <-> Proxy QUIC connection + Datagram flow ID)
(Server-facing socket + Client Connection ID)
    => Client-facing socket
```

Multiple pairs of Connection IDs and sockets can map to the same datagram flow ID or client-facing socket.

These mappings guarantee that any QUIC packet sent from a target to the proxy can be sent to the correct client, in either tunnelled or forwarded mode. Note that this mapping becomes trivial if the proxy always opens a new server-facing socket for every client request with a unique datagram flow ID. The mapping is critical for any case where server-facing sockets are shared or reused.

2.4. Detecting Connection ID Conflicts

In order to be able to route packets correctly in both tunnelled and forwarded mode, proxies check for conflicts before creating a new mapping. If a conflict is detected, the proxy will reject the client's request, as described in Section 5.

Two sockets conflict if and only if all members of the 4-tuple (local IP address, local UDP port, remote IP address, and remote UDP port) are identical.

Two Connection IDs conflict if and only if one Connection ID is equal to or a prefix of another. For example, a zero-length Connection ID conflicts with all connection IDs. This definition of a conflict originates from the fact that QUIC short headers do not carry the length of the Destination Connection ID field, and therefore if two short headers with different Destination Connection IDs are received on a shared socket, one being a prefix of the other prevents the receiver from identifying which mapping this corresponds to.

The proxy treats two mappings as being in conflict when a conflict is detected for all elements on the left side of the mapping diagrams above.

Since very short Connection IDs are more likely to lead to conflicts, particularly zero-length Connection IDs, a proxy MAY choose to reject all requests for very short Connection IDs as conflicts, in anticipation of future conflicts. Note that a request that doesn't contain any Connection ID is equivalent to a request for a zero-length Connection ID, and similarly would cause conflicts when forwarding.

3. Connection ID Headers for CONNECT-UDP

This document defines two headers that can be used in CONNECT-UDP requests and responses. All other requirements defined for CONNECT-UDP [CONNECT-UDP] still apply.

Both "Client-Connection-Id" and "Server-Connection-Id" are Item Structured Headers [STRUCT]. Their value MUST be a Byte Sequence. The byte sequence MAY be zero-length. The byte sequence length MUST NOT exceed 255 bytes. The ABNF is:

```
Client-Connection-Id = sf-binary
Server-Connection-Id = sf-binary
```

"Client-Connection-Id" contains the client connection ID, whereas "Server-Connection-Id" contains the server connection ID.

Like the Datagram-Flow-Id header [CONNECT-UDP], the Client-Connection-Id and Server-Connection-Id headers can only be supported by an HTTP/3 proxy. If a proxy does not support HTTP/3 datagrams [H3DGRAM], or it does not support the extension defined in this document, it MUST NOT send the Client-Connection-Id and Server-Connection-Id headers on any responses. If a proxy does support this extension, it MUST echo the Client-Connection-Id and Server-Connection-Id headers on any 2xx (Successful) responses. Clients that do not receive an echoed Client-Connection-Id or Server-Connection-Id header MUST fall back to using CONNECT-UDP without the extended behavior defined in this document.

4. Client Request Behavior

A client sends new CONNECT-UDP requests when it wants to start a new QUIC connection to a target, when it has received a new Server Connection ID for the target, and before it advertises a new Client Connection ID to the target.

Each request MUST contain a Datagram-Flow-Id header and an authority pseudo-header identifying the target. All requests for the same QUIC

Connection between a client and a target MUST contain the same authority, and SHOULD contain the same Datagram-Flow-Id. If there is Datagram-Flow-Id mismatch, the proxy will treat the requests as different proxied connections, which could appear as a migration or NAT rebinding event to the target.

Each request MUST also contain exactly one connection ID header, either Client-Connection-Id or Server-Connection-Id. Client-Connection-Id requests define paths for receiving packets from the target server to the client, and Server-Connection-Id requests define paths for sending packets from the client to target server.

4.1. New Proxied Connection Setup

The first time that a client uses a proxy for a given QUIC connection, it selects a new datagram flow ID with an even-numbered value [H3DGRAM].

The first request the clients makes MUST contain the authority pseudo-header, the Datagram-Flow-Id header, and the Client-Connection-Id header. These respectively contain the authority of the target, the selected datagram flow ID and the Client Connection ID that will be used in the initial QUIC packets sent through the proxy.

The client can start sending packets tunnelled within DATAGRAM frames as soon as this first CONNECT-UDP request for the datagram flow ID has been sent, even in the same QUIC packet to the proxy. That is, the QUIC packet sent from the client to the proxy MAY contain a STREAM frame containing the CONNECT-UDP request, as well as a DATAGRAM frame that contains a tunnelled QUIC packet to send to the target. This is particularly useful for reducing round trips on connection setup.

Since clients are always aware whether or not they are using a QUIC proxy, clients are expected to cooperate with proxies in selecting Client Connection IDs. A proxy detects a conflict when it is not able to create a unique mapping using the Client Connection ID (Section 2.4). It can reject requests that would cause a conflict and indicate this to the client by replying with a 409 (Conflict) status. In order to avoid conflicts, clients SHOULD select Client Connection IDs of at least 8 bytes in length with unpredictable values. A client also SHOULD NOT select a Client Connection ID that matches the ID used for the QUIC connection to the proxy, as this inherently creates a conflict.

Note that packets sent in DATAGRAM frames before the proxy has sent its CONNECT-UDP response might be dropped if the proxy rejects the request. Specifically, this can occur if the Client Connection ID causes a conflict and the proxy returns a 409 (Conflict) error. Any DATAGRAM frames that are sent in a separate QUIC packet from the STREAM frame that contains the CONNECT-UDP request might also be dropped in the case that the packet arrives at the proxy before the packet containing the STREAM frame.

If the server rejects the first request that uses a specific datagram flow ID, the client MUST retire that datagram flow ID. If the rejection indicated a conflict due to the Client Connection ID, the client MUST select a new Connection ID before sending a new request, and generate a new packet. For example, if a client is sending a QUIC Initial packet and chooses a Connection ID that conflicts with an existing mapping to the same target server, it will need to generate a new QUIC Initial.

4.2. Adding New Client Connection IDs

Since QUIC connection IDs are chosen by the receiver, an endpoint needs to communicate its chosen connection IDs to its peer before the peer can start using them. In QUICv1, this is performed using the NEW_CONNECTION_ID frame.

Prior to informing the target of a new chosen client connection ID, the client MUST send a CONNECT-UDP request with the Client-Connection-Id header to the proxy, and only inform the target of the new client connection ID once a 2xx (Successful) response is received.

4.3. Sending With Forwarded Mode

Once the client has learned the target server's Connection ID, such as in the response to a QUIC Initial packet, it can send a request containing the Server-Connection-Id header to request the ability to forward packets. The client MUST wait for a successful (2xx) response before using forwarded mode. Prior to receiving the server response, the client MUST send short header packets tunnelled in DATAGRAM frames. The client MAY also choose to tunnel some short header packets even after receiving the successful response.

If the client's request that included the Server-Connection-Id is rejected, for example with a 409 (Conflict) response, it MUST NOT forward packets to the requested Server Connection ID, but only use tunnelled mode. The request might also be rejected if the proxy does not support forwarded mode or has it disabled by policy. For any response code other than a 2xx success, the client MUST NOT retry a request for the same Server Connection ID. For errors other than 409 (Conflict), clients SHOULD stop sending requests for other Server Connection IDs in the future.

QUIC long header packets MUST NOT be forwarded. These packets can only be tunnelled within DATAGRAM frames to avoid exposing unnecessary connection metadata.

When forwarding, the client sends a QUIC packet with the target server's Connection ID in the QUIC short header, using the same socket between client and proxy that was used for the main QUIC connection between client and proxy.

4.4. Receiving With Forwarded Mode

A proxy MUST NOT forward packets from the target to the client until after the client has sent at least one packet in forwarded mode. Once this occurs, the proxy MAY use forwarded mode for any Client Connection ID for which it has a valid mapping.

If a client has started using forwarding mode, it MUST be prepared to receive forwarded short header packets on the socket between itself and the proxy for any Client Connection ID that has been accepted by the proxy. The client uses the received Connection ID to determine if a packet was originated by the proxy, or merely forwarded from the target.

4.5. Opting Out of Forwarded Mode

The use of forwarded mode is initiated by the client sending a request with the Server-Connection-Id header and sending at least one forwarded packet to the proxy. A client that does not wish to accept forwarded packets from the proxy when communicating with a specific target can simply not start forwarding packets to the proxy.

5. Proxy Response Behavior

Upon receipt of a CONNECT-UDP request that contains a Client-Connection-Id or Server-Connection-Id header, the proxy validates the request, tries to establish the appropriate mappings as described in Section 2, and establishes a new server-facing socket if necessary.

The proxy MUST validate that the request only contains one of either the Client-Connection-Id or the Server-Connection-Id header, along with a Datagram-Flow-Id header and an authority pseudo-header. If any of these conditions is not met, the proxy MUST reject the request with a 400 (Bad Request) response. The proxy also MUST reject the request if the requested datagram flow ID has already been used on that client <-> proxy QUIC connection with a different requested authority.

The proxy then determines the server-facing socket to associate with the client's datagram flow. This will generally involve performing a DNS lookup for the hostname in the request authority, or finding an existing server-facing socket to the authority. The server-facing socket might already be open due to a previous request from this client, or another. If the socket is not already created, the proxy creates a new one. Proxies can choose to reuse server-facing sockets across multiple datagram flows, or have a unique server-facing socket for every datagram flow.

If a proxy reuses server-facing sockets, it SHOULD store which authorities (which could be a domain name or IP address literal) are being accessed over a particular server-facing socket so it can avoid performing a new DNS query and potentially choosing a different server IP address which could map to a different server.

Server-facing sockets MUST NOT be reused across QUIC and non-QUIC CONNECT-UDP requests, since it might not be possible to correctly demultiplex or direct the traffic. Any packets received on a server-facing socket used for proxying QUIC that does not correspond to a known Connection ID MUST be dropped.

If the request includes a Client-Connection-Id header, the proxy is receiving a request to be able to route traffic back to the client using that Connection ID. If the pair of this Client Connection ID and the selected server-facing socket does not create a conflict, the proxy creates the mapping and responds with a 200 (OK) response. After this point, any packets received by the proxy from the server-facing socket that match the Client Connection ID can to be sent to the client. The proxy MUST use tunnelled mode (DATAGRAM frames) on the correct datagram flow for any long header packets. The proxy SHOULD forward directly to the client for any matching short header packets, but MAY tunnel them in DATAGRAM frames. If the pair is not unique, or the proxy chooses not to support zero-length Client Connection IDs, the proxy responds with a 409 (Conflict) response. If this occurs on the first request for a given datagram flow, the proxy removes any mapping for that datagram flow.

If the request includes a Server-Connection-Id header, the proxy is receiving a request to allow the client to forward packets to the target. If the pair of this Server Connection ID and the client-facing socket on which the request was received does not create a conflict, the proxy creates the mapping and responds with a 200 (OK) response. Once the successful response is sent, the proxy will forward any short header packets received on the client-facing socket that use the Server Connection ID using the correct server-facing socket. If the pair is not unique, the server responds with a 409 (Conflict) response. If this occurs, traffic for that Server Connection ID can only use tunnelled mode, not forwarded.

If the proxy does not support forwarded mode, or does not allow forwarded mode for a particular client or authority by policy, it can reject requests that include the Server-Connection-Id header with a response to indicate the error, such as 403 (Forbidden).

Any successful (2xx) response MUST also echo any Client-Connection-Id, Server-Connection-Id, and Datagram-Flow-Id headers included in the request.

The proxy MUST only forward non-tunnelled packets from the client that are QUIC short header packets (based on the Header Form bit) and have mapped Server Connection IDs. Packets sent by the client that are forwarded SHOULD be considered as activity for restarting QUIC's Idle Timeout [QUIC].

5.1. Removing Mapping State

Each CONNECT-UDP request consumes one bidirectional HTTP/3 stream [HTTP3]. For any stream on which the proxy has sent a response indicating success, any mappings for the request last as long as the stream is open.

A client that no longer wants a given Connection ID to be forwarded by the proxy, for either direction, MUST cancel its CONNECT-UDP HTTP/3 request by closing the corresponding stream.

If the proxy rejects a CONNECT-UDP request by sending a status code of 300 or higher, it MUST close the corresponding stream and remove any associated mappings.

If a client's connection to the proxy is terminated for any reason, all mappings associated with all requests are removed.

A proxy can close its server-facing socket once all datagram flows mapped to that socket have been removed.

5.2. Handling Connection Migration

If a proxy supports QUIC connection migration, it needs to ensure that a migration event does not end up sending too many tunnelled or proxied packets on a new path prior to path validation.

Specifically, the proxy MUST limit the number of packets that it will proxy to an unvalidated client address to the size of an initial congestion window. Proxies additionally SHOULD pace the rate at which packets are sent over a new path to avoid creating unintentional congestion on the new path.

6. Example

Consider a client that is establishing a new QUIC connection through the proxy. It has selected a Client Connection ID of 0x31323334. It selects the next open datagram flow ID (2). In order to inform a proxy of the new QUIC Client Connection ID, and binds that connection ID to datagram flow ID 2, the client sends the following CONNECT-UDP request:

```
HEADERS
:method = CONNECT-UDP
:authority = target.example.com:443
client-connection-id = :MTIzNA==:
datagram-flow-id = 2
```

The client will also send the initial QUIC packet with the Long Header form in a DATAGRAM frame with flow ID 2.

Once the proxy sends a 200 response indicating success, packets received by the proxy that match the Connection ID 0x31323334 will be directly forwarded to the client. The proxy will also forward the initial QUIC packet received on DATAGRAM flow ID 2 to target.example.com:443.

When the proxy receives a response from target.example.com:443 that has 0x31323334 as the Destination Connection ID, the proxy will forward that packet to the client on DATAGRAM flow ID 2.

Once the client learns which Connection ID has been selected by the target server, it can send a new request to the proxy to establish a mapping. In this case, that ID is 0x61626364. The client sends the following request:

```
HEADERS
:method = CONNECT-UDP
:authority = target.example.com:443
server-connection-id = :YWJjZA==:
datagram-flow-id = 2
```

The client also sends its reply to the target server in a DATAGRAM frame on flow ID 2 after sending the new request.

Once the proxy sends a 200 response indicating success, packets sent by the client that match the Connection ID 0x61626364 will be forwarded to the target server, i.e., without proxy decryption.

Upon receiving the response, the client starts sending Short Header packets with a Destination Connection ID of 0x61626364 directly to the proxy (not tunnelled), and these are forwarded directly to the target by the proxy. Similarly, Short Header packets from the target with a Destination Connection ID of 0x31323334 are forwarded directly to the client.

7. Interactions with Load Balancers

Some QUIC servers are accessed using load balancers, as described in [QUIC-LB]. These load balancers route packets to servers based on the server's Connection ID. These Connection IDs are generated in a way that can be coordinated between servers and their load balancers.

If a proxy that supports this extension is itself running behind a load balancer, extra complexity arises once clients start using forwarding mode and sending packets to the proxy that have Destination Connection IDs that belong to the end servers, not the proxy. If the load balancer is not aware of these Connection IDs, or the Connection IDs conflict with other Connection IDs used by the load balancer, packets can be routed incorrectly.

QUIC-aware CONNECT-UDP proxies that use forwarding mode generally SHOULD NOT be run behind load balancers; and if they are, they MUST coordinate between the proxy and the load balancer to create mappings for proxied Connection IDs prior to the proxy sending 2xx (Successful) responses to clients.

QUIC-aware CONNECT-UDP proxies that do not allow forwarding mode can function unmodified behind QUIC load balancers.

8. Packet Size Considerations

Since Initial QUIC packets must be at least 1200 bytes in length, the DATAGRAM frames that are used for a QUIC-aware CONNECT-UDP proxy MUST be able to carry at least 1200 bytes.

Additionally, clients that connect to a proxy for purpose of proxying QUIC SHOULD start their connection with a larger packet size than 1200 bytes, to account for the overhead of tunnelling an Initial QUIC packet within a DATAGRAM frame. If the client does not begin with a larger packet size than 1200 bytes, it will need to perform Path MTU (Maximum Transmission Unit) discovery to discover a larger path size prior to sending any tunnelled Initial QUIC packets.

Once a proxied QUIC connections moves into forwarded mode, the client SHOULD initiate Path MTU discovery to increase its end-to-end MTU.

9. Security Considerations

Proxies that support this extension SHOULD provide protections to rate-limit or restrict clients from opening an excessive number of proxied connections, so as to limit abuse or use of proxies to launch Denial-of-Service attacks.

Sending QUIC packets by forwarding through a proxy without tunnelling exposes some QUIC header metadata to onlookers, and can be used to correlate packets flows if an attacker is able to see traffic on both sides of the proxy. Tunnelled packets have similar inference problems. An attacker on both sides of the proxy can use the size of ingress and egress packets to correlate packets belonging to the same connection. (Absent client-side padding, tunneled packets will typically have a fixed amount of overhead that is removed before their DATAGRAM contents are written to the target.)

Since proxies that forward QUIC packets do not perform any cryptographic integrity check, it is possible that these packets are either malformed, replays, or otherwise malicious. This may result in proxy targets rate limiting or decreasing the reputation of a given proxy.

10. IANA Considerations

10.1. HTTP Headers

This document registers the "Client-Connection-Id" and "Server-Connection-Id" headers in the "Permanent Message Header Field Names" <https://www.iana.org/assignments/message-headers>.

Header Field Name	Protocol	Status	Reference
Client-Connection-Id	http	exp	This document
Server-Connection-Id	http	exp	This document

11. References

11.1. Normative References

[CONNECT-UDP] Schinazi, D., "The CONNECT-UDP HTTP Method", Work in Progress, Internet-Draft, draft-ietf-masque-connect-udp-03, 5 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-masque-connect-udp-03.txt>.

[DGRAM] Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-datagram-01, 24 August 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-datagram-01.txt>.

[H3DGRAM] Schinazi, D., "Using QUIC Datagrams with HTTP/3", Work in Progress, Internet-Draft, draft-schinazi-quic-h3-datagram-05, 12 October 2020, <http://www.ietf.org/internet-drafts/draft-schinazi-quic-h3-datagram-05.txt>.

[HTTP3] Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-33, 15 December 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-http-33.txt>.

[INVARIANTS] Thomson, M., "Version-Independent Properties of QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-invariants-13, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-invariants-13.txt>.

[QUIC] Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", Work in Progress, Internet-Draft, draft-ietf-quic-transport-34, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-transport-34.txt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[STRUCT] Nottingham, M. and P. Kamp, "Structured Field Values for HTTP", Work in Progress, Internet-Draft, draft-ietf-httpbis-header-structure-19, 3 June 2020, <http://www.ietf.org/internet-drafts/draft-ietf-httpbis-header-structure-19.txt>.

11.2. Informative References

[QUIC-LB] Duke, M. and N. Banks, "QUIC-LB: Generating Routable QUIC Connection IDs", Work in Progress, Internet-Draft, draft-ietf-quic-load-balancers-05, 30 October 2020, <http://

www.ietf.org/internet-drafts/draft-ietf-quic-load-balancers-05.txt>.

Acknowledgments

Thanks to Lucas Pardue, Ryan Hamilton, and Mirja Kuehlewind for their inputs on this document.

Authors' Addresses

Tommy Pauly
Apple Inc.
One Apple Park Way
Cupertino, California 95014,
United States of America

Email: tpauly@apple.com

David Schinazi
Google LLC
1600 Amphitheatre Parkway
Mountain View, California 94043,
United States of America

Email: dschinazi.ietf@gmail.com