MASQUE
Internet-Draft
Intended status: Standards Track
Expires: 23 April 2026

M. Westerlund
Ericsson
M. Seemann
Smallstep
M. K端hlewind
M. Ihlar
Ericsson

20 October 2025

# ECN and DSCP support for HTTPS's Connect-UDP
# draft-westerlund-masque-connect-udp-ecn-dscp-01

## Abstract

HTTP's Extended Connect's Connect-UDP protocol enables a client to proxy a UDP flow from the HTTP server towards a specified target IP address and UDP port. QUIC and Real-time transport protocol (RTP) are examples of transport protocols that use UDP and support Explicit Congestion Notification (ECN) and provide the necessary feedback. This document specifies how ECN and DSCP can be supported through an extension to the Connect-UDP protocol for HTTP.

## About This Document

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

The latest revision of this draft can be found at https://gloinul.github.io/masque-ecn/#go.draft-westerlund-masque-connect-udp-ecn.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-westerlund-masque-connect-udp-ecn-dscp/.

Discussion of this document takes place on the MASQUE Working Group mailing list (mailto:masque@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/masque/. Subscribe at https://www.ietf.org/mailman/listinfo/masque/.

Source for this draft and an issue tracker can be found at https://github.com/gloinul/masque-ecn.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 23 April 2026.

## Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

## 1. Introduction

Connect-UDP, as currently defined, limits the Explicit Congestion Notification (ECN) [RFC3168] exchange between the HTTP server and the target. There is no support for carrying the ECN bits between the HTTP Connect-UDP client and the HTTP server proxying the UDP flow. Thus, it is not possible to establish the end-to-end ECN information flow necessary to support either classic ECN [RFC3168] or L4S [RFC9330], [RFC9331].

Diffserv [RFC2475] enables differential network treatment of packets. Connect-UDP, as currently defined, lacks support for carrying the DSCP field [RFC2474] through the tunnel.

This document specifies two Connect-UDP extensions that enable end-to-end ECN and DSCP for proxied connections: an ECN-zero-bytes extension, which adds no overhead by encoding the ECN value directly into the Context ID; and an ECN/DSCP extension, which carries both DSCP and ECN in the HTTP Datagram payload. For these two extensions, this document specifies negotiation and defines a new Datagram context that carries the DSCP and ECN bits and the UDP payload, replacing the context defined in [RFC9298].

An alternative to this extension is Connect-IP [RFC9484]; however, it carries a full IP header between the HTTP client and server, resulting in significantly more overhead than this extension, which requires zero or one byte to carry both the DSCP and ECN bits.

To define a solution that can be combined with other extensions, and thus other contexts, without redefining each combination, the extensions defined in this document indicate not only the ECN and DSCP values but also the next Context ID. The ECN-zero-bytes extension defines three additional Context ID values that are bound to an HTTP Datagram payload identifying the Context ID and indicate whether the packet was marked with ECT(0), ECT(1), or CE, respectively.

The extensions are defined such that they allow clients to optimistically start sending UDP packets in HTTP Datagrams, i.e. before receiving the response to its UDP proxying request, as described in Section 5 of [RFC9298].

An endpoint should not enable both extensions defined in this document, as that would lead to confusion if both extensions indicate different ECN values.

## 2. Conventions

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.

## 3. ECN-zero-byte Extension

For a zero-overhead encoding, the ECN bits can be indicated by using different Context IDs. At the same time, the Context ID indicates which Context ID would otherwise have been used to identify the structure of the rest of the HTTP payload, which we call the payload-identifying context ID, or short payload ID. Often this is the basic UDP-payload context (Context ID 0) as defined by [RFC9298].

The core of this solution is to define additional Context IDs (A, B, and C) for a Connect-UDP stream that indicate ECN values other than Not-ECT, i.e., ECT(1), ECT(0), or CE [RFC3168]. This idea is shown in Table 1.

| Context ID Value | ECN bit | ECN Value | Payload ID |
|---|---|---|---|
| 0 | 0b00 | Not-ECT | 0 |
| A | 0b01 | ECT(1) | 0 |
| B | 0b10 | ECT(0) | 0 |
| C | 0b11 | CE | 0 |

Table 1: ECN Encoding Table

No new Context ID value is defined to represent Not-ECT, since using a Context ID without this extension would, by default, imply Not-ECT. Additionally, Context IDs are defined to represent the combination of

an ECN value other than Not-ECT and the payload-identifying Context ID. If an application uses more Context ID values than just zero, additional Context IDs must be defined.

This extension results in four times as many Context IDs within a single Connect-UDP stream. We expect that this is acceptable in most cases, as a total of 31 client initiated Context IDs can be encoded in a single byte, thus resulting in no packet expansion. However, for applications that have more than 8 original Context IDs (including zero), it is recommended to use the ECN/DSCP extension Section 4, which only doubles the number of Context IDs but requires an additional byte in the payload.

An endpoint enabling this extension MUST define all three ECN values, even if the ECN-enabled application expects that only one ECT value (and CE) is used. This is because of transmission errors or erroneous remarking in the network, where the other ECT codepoint, as well as Not-ECT, may be observed.

Negotiation of the conetxt ID values is defined using both HTTP headers and capsulses in Section 5.1.

## 4. DSCP plus ECN Extension

The HTTP Datagram Payload format is defined in [RFC9484] as depicted below.

```
UDP Proxying HTTP Datagram Payload {
  Context ID (i),
  UDP Proxying Payload (..)
}
```

Figure 1: ECN enabled UDP Proxying HTTP Datagram Format

For the DSCP plus ECN extension, the DSCP plus ECN UDP Proxying Payload is defined in the following way:

```
ECN/DSCP UDP Proxying Payload {
  DSCP(6),
  ECN(2),
  UDP Proxying Payload (..) # Payload per another Context ID definition
}
```

Figure 2: DSCP plus ECN UDP Proxying Payload

DSCP: A six bit field carrying the Differentiated Service Code Point as defined by [RFC2474] associated with this UDP Proxying Payload.

ECN: A two bit field carrying the IP ECN bits as defined by Section 5 of [RFC3168] to be set or as received on the IP/UDP packet carrying the UDP Datagram payload included.

UDP Proxying Payload: Another UDP Proxying Payload following the ECN carrying byte. This uses another Context ID as negotiated, e.g. Context ID 0. Thus enabling this byte to be combined with any other payload.

This format used a negotiated context ID that MUST be non-zero. It MUST also negotiate the payload-identifying context ID.

## 5. Negotiating Extensions Usage

This section defines capability negotation and Context ID configuration for the two defined extensions.

Note that Context Identifiers are defined as QUIC varints (see Section 16 of [RFC9000]) and support values up to 4,611,686,018,427,387,903 (2^62-1), which is larger than what a Structure Header Integer supports (limited to 999,999,999,999,999). We foresee no issues with this limitation, as Context Identifiers should primarily use the single-byte representation for efficiency, i.e., they should rarely exceed 63.

### 5.1. ECN-zero-byte Extension

To use the ECN-zero-byte extension three Context IDs need to be configured relative to the Context ID that identifies the actual HTTP Datagram payload.

A configuration of ECN signaling is represented by a four-tuple with the following format:

```
ECN_CONTEXT_ASSIGNMENT {
  ECT_1_CONTEXT (i),
  ECT_0_CONTEXT (i),
  CE_CONTEXT (i),
  PAYLOAD_CONTEXT (i)
}
```

Figure 3: ECN CONTEXT ASSIGNMENT Format

ECT_1_CONTEXT: The Context ID used to indicate the payload was marked with ECN value ECT(1).

ECT_0_CONTEXT: The Context ID used to indicate the payload was marked with ECN value ECT(0).

CE_CONTEXT:  The Context ID used to indicate the payload was marked with ECN value CE.

PAYLOAD_CONTEXT:  The payload-identifiying context ID indicating the actual encoding of the start of the HTTP Datagram payload.

#### 5.1.1. HTTP Structured field

ECN-Context-ID is a Structured Header Field [RFC9651]. Its value is a List consisting of zero or more Inner Lists, where the Inner List contains four Integer Items. The Integer Items MUST be non-negative, as they are context identifiers defined in [RFC9298]. The four Context IDs are those defined in Figure 3, in that order.

When the header is included in an Extended Connect request, it indicates, first of all, support for this ECN extension. Secondly, it may define one or more 4-item inner lists of Context IDs for the requestor-to-responder direction. If no 4-item inner lists of Context IDs are included, then this header only indicates support for the extension, and the Context IDs MAY be signaled using capsules.

When the request includes the ECN-Context-ID header, the responder MAY include this header in the response. If included with one or more 4-item inner lists, it defines Context ID defined by the server, usable in either direction.

The following example indicates support for ECN-zero-byte-extension and defines two sets of client initiated Context IDs: ID=10, 12, 14 (ECT(1), ECT(0), CE) combined with Context ID 8; and ID=2, 4, 6 combined with the default ID=0 as defined in [RFC9298], i.e., a plain UDP payload.

```
ECN-Context-ID: (10,12,14,8), (2,4,6,0)
```

Figure 4: Example of ECN-Context-ID header

#### 5.1.2. ECN Context ID Assignment Capsule

The ECN_CONTEXT_ASSIGN capsule is used to assign Context ID values to the ECN-zero-byte extension.

```
ECN_CONTEXT_ASSIGN Capsule {
  Type (i) = TBA_2
  Length (i),
  ECN_CONTEXT_ASSIGNMENT (..) ...,
}
```

Figure 5: ECN_CONTEXT_ASSIGN Capsule Format

Type and Length as defined by Section 3.2 of the HTTP Capsule specification [RFC9297]. The capsule value is the ECN_CONTEXT_ASSIGNMENT defined above in Figure 3. Thus, the capsule value consists of zero or more ECN_CONTEXT_ASSIGNMENT four-tuples.

## 5.2. DSCP plus ECN extension

This section defines the negotiation of the DSCP plus ECN extension defined in Section 4. It defines both an HTTP header field (DSCP-ECN-Context-ID) that can be included in the Extended Connect request, and a capsule.

### 5.2.1. HTTP Structured Header

DSCP-ECN-Context-ID is a Structured Header Field [RFC9651]. Its value is a List of zero or more Inner Lists, where each Inner List contains two Integer Items. The Integer Items MUST be non-negative, as they are context identifiers defined by [RFC9298]. The first Integer Item is the Context ID being defined, and the second Integer Item is the Context ID for the payload following the DSCP plus ECN byte.

When the header is included in an Extended Connect request, it indicates, first of all, support for the DSCP plus ECN extension. Secondly, it may define one or more context ID pairs define by the client. If no context ID pairs are included, then this header only indicates support for the extension, and it may be configured using capsules.

When the request includes the DSCP-ECN-Context-ID header, the responder MAY include this header in the response. If included, it defines server initated Context ID(s) if one or more Inner Lists are included.

The following example indicates supoprt of the ECN/DSCP extension and defines three Context IDs: Context ID 10 combined with Context ID 8, Context ID 6 combined with Context ID 4, and Context ID 2 combined with the default Context ID 0 as defined in [RFC9298], i.e., a plain UDP payload.

DSCP-ECN-Conext-ID: (10,8), (6,4), (2,0)

Figure 6: Example of ECN-Context-ID header

### 5.2.2. DSCP plus ECN Context ID Assignment Capsule

The DSCP_ECN_CONTEXT_ASSIGN capsule is used to assign context ID values for the DSCP/ECN extension.

```
DSCP_ECN_CONTEXT_ASSIGN Capsule {
  Type (i) = TBA_2
  Length (i),
  CONTEXT ASSIGNMENT (..) ...,
}
```

Figure 7: DSCP_ECN_CONTEXT_ASSIGN Capsule Format

Type and Length as defined by the HTTP Capsule specification in Section 3.2 of [RFC9297]. The capsule value is defined below.

```
CONTEXT ASSIGNMENT {
  ASSIGNED_CONTEXT (i),
  NEXT_PAYLOAD_CONTEXT (i)
}
```

Figure 8: CONTEXT ASSIGNMENT Format

The capsule value consists of zero or more CONTEXT ASSIGNMENT pair values. Each pair consists of these two fields:

ASSIGNED_CONTEXT: : The context ID identifying that the indicated HTTP datagram payload starts with the DSCP plus ECN UDP Proxying Payload.

NEXT_PAYLOAD_CONTEXT: The context ID identifying the following payload in each HTTP datagram after the DSCP plus ECN UDP Proxying Payload.

This capsule is sent by either endpoints to configure or extend the configuration of context IDs.

## 6. Tunnels and DSCP and ECN marking interactions

### 6.1. Tunnel Endpoint Marking

The Tunnel Endpoint, when receiving an IP/UDP packet belonging to a Connect-UDP request with the DSCP plus ECN extension enabled, copies the six DSCP bits and the two ECN bits from the IP header into the DSCP and ECN fields, respectively, in the HTTP Datagram payload using the relevant Context ID. When using the ECN-zero-byte extension, the two ECN bits in the incoming IP/UDP packet are used to select the appropriate Context ID.

For the DSCP plus ECN extension, the Tunnel Endpoint on egress copies the DSCP and ECN extension field values into the IP/UDP packet it creates for this UDP Proxying payload. For the ECN-zero-byte extension, the Context ID value is used to determine which value to write in the outgoing IP/UDP packet寰洀 ECN field.

A Tunnel endpoint which is unable to read or set the ECN Field SHALL NOT enable the ECN extension.

## 6.2. DSCP Remarking Considerations

The tunnel may interconnect two different administrative domains that use DSCP values differently. Thus, the endpoints likely need to perform remarking of DSCP field values, similar to what an inter-domain router would. Depending on use cases and deployment, the HTTP client can be in different network domains with different DSCP usages. An HTTP server that, based on user identification, connects the HTTP client to different network domains behind it may also need to support multiple external domains.

The above complications in handling DSCP make it impossible to provide a standardized remarking instruction. Instead, the deployment will have to define whether remarking is handled by the HTTP server, the HTTP client, or both, considering the tunnel a specific network domain in itself.

## 6.3. Tunnel Transport Connection ECN Interactions and Congestion Control

The primary goal of the ECN extension is to enable ECN usage between the proxy and the target and to have the end-to-end transport react to that ECN. However, different potential models exist for providing ECN interactions for the tunnel, i.e., between the HTTP client and server. The choice depends on how the tunnel is configured and what additional support has been implemented for the Connect-UDP protocol.

The default deployment would be to use congestion controlled transport protocols between the HTTP endpoints or proxies for the tunneled ECN enabled packets. This include all HTTP versions before HTTP/3 [RFC9114], as well as HTTP/3 sending packets over reliable streams as well as over congestion controlled datagrams. In this deployment on the ingress to each congestion controlled transport an Active Queue Management (AQM) is recommend that can mark the tunneled packet's ECN field (or drop them) in case there is sufficient queue build up.

For tunnels using HTTP/3 with datagrams, where the QUIC connection disables congestion control on packets containing HTTP datagrams as discussed in Section 6 of [RFC9298], the ECN marking on tunneled packets can be propagated between the IP packet of the transport connection and the end-to-end packet. This represents a specific implementation of IP-in-IP tunnels with tightly coupled shim headers as discussed in [RFC9601]. It is implemented as Feed-Forward-and-Up as discussed in [RFC9599], and MUST use the normal mode on tunnel ingress and follow the specified default behavior on egress as defined in [RFC6040].

### 6.4. Tunnel Transport Connection DSCP Interactions

For HTTP tunnels not using HTTP/3 [RFC9114], HTTP/3 using reliable streams, or HTTP/3 with datagrams but without disabling congestion control, the tunnel will consist of one or possibly several chained congestion-controlled transport connections. These transport connections can use only a single DSCP code point to avoid inconsistent network treatment that might confuse the congestion controller and retransmission mechanism. Thus, even if the tunneled packets use different DSCP values, the transport connection must settle on a single, suitable DSCP value. However, if the QUIC multipath extension [I-D.ietf-quic-multipath] is used, each path can have a different DSCP value. In this latter case, packets with different DSCP values can be mapped to different paths with the appropriate network treatment as indicated by their DSCP values.

For tunnels using HTTP/3 with datagrams and where the QUIC connection disables congestion control on packets containing HTTP datagrams, as discussed in Section 6 of [RFC9298], the QUIC packets can be marked using the most suitable DSCP value based on the encapsulated packet. In cases where the tunnel connection is sent into a different network domain than the one on which the tunneled packet was received, a suitable remapping must occur for the domain to which the tunnel packet will be sent. The HTTP tunnel MUST NOT coalesce different tunneled payloads that are not mapped to the same DSCP in a single QUIC packet.

### 6.5. QUIC Aware Forwarding

An HTTP endpoint that supports this extension and QUIC Aware Forwarding [I-D.ietf-masque-quic-proxy] MUST preserve ECN markings on forwarded packets in both directions to ensure end-to-end ECN functionality. Using this extension in combination with QUIC Aware Forwarding, rather than relying solely on the latter, also ensures that ECN black holes do not occur, for example, on long-header packets or packets sent before the QUIC Aware Forwarding path is established for short-header packets. Thus, supporting both provides

a consistent ECN experience.

# 7. IANA Considerations

## 7.1. HTTP Field Names

IANA is request to register two new permanent Field name in the Hypertext Transfer Protocol (HTTP) Field Name Registry (At time of writing residing at: https://www.iana.org/assignments/http-fields/http-fields.xhtml).

### 7.1.1. ECN-Context-ID

Field Name: ECN-Context-ID

Status: Permanent

Structured Type: List

Reference: [RFC-TO-BE]

### 7.1.2. DSCP-ECN-Context-ID

Field Name: DSCP-ECN-Context-ID

Status: Permanent

Structured Type: List

Reference: [RFC-TO-BE]

## 7.2. HTTP Capsule Type

IANA is reqeusted ot register two new HTTP Capsule Types in the permanent range (0x00-0x3f).

### 7.2.1. ECN_CONTEXT_ASSIGN

Value:  TBA_1

Capsule Type:  ECN_CONTEXT_ASSIGN

Status:  permanent

Reference:  RFC-TO-BE

Change Controller:  IETF

Contact:  MASQUE Working Group masque@ietf.org

Notes:  None

#### 7.2.2.  DSCP_ECN_CONTEXT_ASSIGN

Value:  TBA_2

Capsule Type:  DSCP_ECN_CONTEXT_ASSIGN

Status:  permanent

Reference:  RFC-TO-BE

Change Controller:  IETF

Contact:  MASQUE Working Group masque@ietf.org

Notes:  None

## 8.  Acknowledgments

This draft takes insperation from David Schinazi's An ECN Extension to Connect-UDP [I-D.schinazi-masque-connect-udp-ecn].

## 9.  References

### 9.1.  Normative References

[I-D.ietf-masque-quic-proxy]
Pauly, T., Rosenberg, E., and D. Schinazi, "QUIC-Aware Proxying Using HTTP", Work in Progress, Internet-Draft, draft-ietf-masque-quic-proxy-07, 8 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-masque-quic-proxy-07>.

[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/rfc/rfc2119>.

[RFC2474]  Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/rfc/rfc2474>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/rfc/rfc3168>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <https://www.rfc-editor.org/rfc/rfc6040>.

[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/rfc/rfc8174>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

[RFC9114] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/rfc/rfc9114>.

[RFC9297] Schinazi, D. and L. Pardue, "HTTP Datagrams and the Capsule Protocol", RFC 9297, DOI 10.17487/RFC9297, August 2022, <https://www.rfc-editor.org/rfc/rfc9297>.

[RFC9298] Schinazi, D., "Proxying UDP in HTTP", RFC 9298, DOI 10.17487/RFC9298, August 2022, <https://www.rfc-editor.org/rfc/rfc9298>.

[RFC9331] De Schepper, K. and B. Briscoe, Ed., "The Explicit Congestion Notification (ECN) Protocol for Low Latency, Low Loss, and Scalable Throughput (L4S)", RFC 9331, DOI 10.17487/RFC9331, January 2023, <https://www.rfc-editor.org/rfc/rfc9331>.

[RFC9599] Briscoe, B. and J. Kaippallimalil, "Guidelines for Adding Congestion Notification to Protocols that Encapsulate IP", BCP 89, RFC 9599, DOI 10.17487/RFC9599, August 2024, <https://www.rfc-editor.org/rfc/rfc9599>.

[RFC9601] Briscoe, B., "Propagating Explicit Congestion Notification across IP Tunnel Headers Separated by a Shim", RFC 9601, DOI 10.17487/RFC9601, August 2024, <https://www.rfc-editor.org/rfc/rfc9601>.

[RFC9651] Nottingham, M. and P. Kamp, "Structured Field Values for HTTP", RFC 9651, DOI 10.17487/RFC9651, September 2024, <https://www.rfc-editor.org/rfc/rfc9651>.

## 9.2. Informative References

[I-D.ietf-quic-multipath]
Liu, Y., Ma, Y., De Coninck, Q., Bonaventure, O., Huitema, C., and M. Kテシhlewind, "Managing multiple paths for a QUIC connection", Work in Progress, Internet-Draft, draft-ietf-quic-multipath-17, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-multipath-17>.

[I-D.schinazi-masque-connect-udp-ecn]
Schinazi, D., "An ECN Extension to CONNECT-UDP", Work in Progress, Internet-Draft, draft-schinazi-masque-connect-udp-ecn-02, 28 March 2022, <https://datatracker.ietf.org/doc/html/draft-schinazi-masque-connect-udp-ecn-02>.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <https://www.rfc-editor.org/rfc/rfc2475>.

[RFC8311] Black, D., "Relaxing Restrictions on Explicit Congestion Notification (ECN) Experimentation", RFC 8311, DOI 10.17487/RFC8311, January 2018, <https://www.rfc-editor.org/rfc/rfc8311>.

[RFC9330] Briscoe, B., Ed., De Schepper, K., Bagnulo, M., and G. White, "Low Latency, Low Loss, and Scalable Throughput (L4S) Internet Service: Architecture", RFC 9330, DOI 10.17487/RFC9330, January 2023, <https://www.rfc-editor.org/rfc/rfc9330>.

[RFC9484] Pauly, T., Ed., Schinazi, D., Chernyakhovsky, A., Kテシhlewind, M., and M. Westerlund, "Proxying IP in HTTP", RFC 9484, DOI 10.17487/RFC9484, October 2023, <https://www.rfc-editor.org/rfc/rfc9484>.

## Authors' Addresses

Magnus Westerlund
Ericsson
Email: magnus.westerlund@ericsson.com

Marten Seemann
Smallstep
Email: martenseemann@gmail.com

Mirja Kテシhlewind
Ericsson
Email: mirja.kuehlewind@ericsson.com

Marcus Ihlar
Ericsson
Email: marcus.ihlar@ericsson.com