Workgroup: QUIC Working Group
Internet-Draft: draft-ietf-quic-multipath-04
Published: 13 March 2023
Intended Status: Standards Track
Expires: 14 September 2023
A   Y. Liu, Ed.   Y. Ma   Q. De Coninck, Ed.
        uAlibaba Inc.   Alibaba Inc.   UCLouvain
        t
        h
        o
        r
        s
        :
         O. Bonaventure   C. Huitema
         UCLouvain and Tessares   Private Octopus Inc.
         M. Kuehlewind, Ed.
         Ericsson

Multipath Extension for QUIC

Abstract

This document specifies a multipath extension for the QUIC protocol to enable the simultaneous usage of multiple paths for a single connection.

Discussion Venues

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

Discussion of this document takes place on the QUIC Working Group mailing list (quic@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/quic/.

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

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 14 September 2023.

Copyright Notice

Copyright (c) 2023 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

This document specifies an extension to QUIC version 1 [QUIC-TRANSPORT] to enable the simultaneous usage of multiple paths for a single connection.

This proposal is based on several basic design points:

- Re-use as much as possible mechanisms of QUIC version 1. In particular, this proposal uses path validation as specified for QUIC version 1 and aims to re-use as much as possible of QUIC's connection migration.

- Use the same packet header formats as QUIC version 1 to avoid the risk of packets being dropped by middleboxes (which may only support QUIC version 1)

- Congestion Control must be per-path (following [QUIC-TRANSPORT]) which usually also requires per-path RTT measurements

- PMTU discovery should be performed per-path

- The use of this multipath extension requires the use of non-zero Connection IDs in both directions.

- A path is determined by the 4-tuple of source and destination IP address as well as source and destination port. Therefore, there can be at most one active paths/connection ID per 4-tuple.

- If the 4-tuple changes without the use of a new connection ID (e.g. due to a NAT rebinding), this is considered as a migration event.

The path management specified in Section 9 of [QUIC-TRANSPORT] fulfills multiple goals: it directs a peer to switch sending through a new preferred path, and it allows the peer to release resources associated with the old path. Multipath requires several changes to that mechanism:

- Allow simultaneous transmission of non-probing frames on multiple paths.

- Continue using an existing path even if non-probing frames have been received on another path.

- Manage the removal of paths that have been abandoned.

As such, this extension specifies a departure from the specification of path management in Section 9 of [QUIC-TRANSPORT] and therefore requires negotiation between the two endpoints using a new transport parameter, as specified in Section 3.

This extension uses multiple packet number spaces. When multipath is negotiated, each destination connection ID is linked to a separate packet number space. Using multiple packet number spaces enables

direct use of the loss recovery and congestion control mechanisms defined in [QUIC-RECOVERY].

This specification requires the sender to use a non-zero connection ID when opening an additional path. Some deployments of QUIC use zero-length connection IDs. However, when a node selects to use zero-length connection IDs, it is not possible to use different connection IDs for distinguishing packets sent to that node over different paths.

Each endhost may use several IP addresses to serve the connection. In particular, the multipath extension supports the following scenarios.

* The client uses multiple IP addresses and the server listens on only one.

* The client uses only one IP address and the server listens on several ones.

* The client uses multiple IP addresses and the server listens on several ones.

This proposal does not cover address discovery and management. Addresses and the actual decision process to setup or tear down paths are assumed to be handled by the application that is using the QUIC multipath extension. This is sufficient to address the first aforementioned scenario. However, this document does not prevent future extensions from defining mechanisms to address the remaining scenarios. Further, this proposal only specifies a simple basic packet scheduling algorithm, in order to provide some basic implementation guidance. However, more advanced algorithms as well as potential extensions to enhance signaling of the current path state are expected as future work.

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.

We assume that the reader is familiar with the terminology used in [QUIC-TRANSPORT]. When this document uses the term "path", it refers to the notion of "network path" used in [QUIC-TRANSPORT].

2. High-level overview

The multipath extensions to QUIC proposed in this document enable the simultaneous utilization of different paths to exchange non-probing QUIC frames for a single connection. This contrasts with the base QUIC protocol [QUIC-TRANSPORT] that includes a connection migration mechanism that selects only one path to exchange such frames.

A multipath QUIC connection starts with a QUIC handshake as a regular QUIC connection. See further Section 3. The peers use the

enable_multipath transport parameter during the handshake to negotiate the utilization of the multipath capabilities. The active_connection_id_limit transport parameter limits the maximum number of active paths that can be used during a connection. A multipath QUIC connection is thus an established QUIC connection where the enable_multipath transport parameter has been successfully negotiated.

To add a new path to an existing multipath QUIC connection, a client starts a path validation on the chosen path, as further described in Section 4. In this version of the document, a QUIC server does not initiate the creation of a path, but it can validate a new path created by a client. A new path can only be used once the associated 4-tuple has been validated by ensuring that the peer is able to receive packets at that address (see Section 8 of [QUIC-TRANSPORT]). The Destination Connection ID is used to associate a packet to a packet number space that is used on a valid path. Further, the sequence number of Destination Connection ID is used as numerical identifier in control frames. E.g. an endpoint sends a PATH_ABANDON frame to request its peer to abandon the path on which the sender uses the Destination Connection ID with the sequence number contained in the PATH_ABANDON frame.

In addition to these core features, an application using Multipath QUIC will typically need additional algorithms to handle the number of active paths and how they are used to send packets. As these differ depending on the application's requirements, their specification is out of scope of this document.

Using multiple packet number spaces requires changes in the way AEAD is applied for packet protection, as explained in Section 5.2, and tighter constraints for key updates, as explained in Section 5.3.

3. Handshake Negotiation and Transport Parameter

This extension defines a new transport parameter, used to negotiate the use of the multipath extension during the connection handshake, as specified in [QUIC-TRANSPORT]. The new transport parameter is defined as follows:

*name: enable_multipath (current version uses 0x0f739bbc1b666d04)

*value: 0 (default) for disabled.

The valid options for the value field are listed in Table 1:

Option	Definition
0x0	don't support multipath
0x1	supports multipath as defined in this document

Table 1: Available value for enable_multipath

If for any one of the endpoints, the parameter is absent or set to 0, the endpoints MUST fallback to [QUIC-TRANSPORT] with single active path and MUST NOT use any frame or mechanism defined in this document.

If the parameter is set to 1, both endpoints MUST use non-zero connection IDs. If an enable_multipath parameter set to 1 is received and the carrying packet does not contain a non-zero length connection ID, the receiver MUST treat this as a connection error of type TRANSPORT_PARAMETER_ERROR (specified in Section 20.1 of [QUIC-TRANSPORT]) and close the connection.

If endpoint receives an unexpected value for the transport parameter "enable_multipath", it MUST treat this as a connection error of type TRANSPORT_PARAMETER_ERROR (specified in Section 20.1 of [QUIC-TRANSPORT]) and close the connection.

This extension does not change the definition of any transport parameter defined in Section 18.2. of [QUIC-TRANSPORT].

Inline with the definition in [QUIC-TRANSPORT] disable_active_migration also disables multipath support, except "after a client has acted on a preferred_address transport parameter" (Section 18.2. of [QUIC-TRANSPORT]).

The transport parameter "active_connection_id_limit" [QUIC-TRANSPORT] limits the number of usable Connection IDs, and also limits the number of concurrent paths. For the QUIC multipath extension this limit even applies when no connection ID is exposed in the QUIC header.

4. Path Setup and Removal

After completing the handshake, endpoints have agreed to enable multipath feature and can start using multiple paths. This document does not specify how an endpoint that is reachable via several addresses announces these addresses to the other endpoint. In particular, if the server uses the preferred_address transport parameter, clients SHOULD NOT assume that the initial server address and the addresses contained in this parameter can be simultaneously used for multipath. Furthermore, this document does not discuss when a client decides to initiate a new path. We delegate such discussion in separate documents.

To open a new path, an endpoint SHALL use different Connection IDs on different paths. Still, the receiver may observe the same Connection ID used on different 4-tuples due to, e.g., NAT rebinding. In such case, the receiver reacts as specified in Section 9.3 of [QUIC-TRANSPORT].

This proposal adds two multipath control frames for path management:

- PATH_ABANDON frame for the receiver side to abandon the path (see Section 8.2)

- PATH_STATUS frame to express a preference in path usage (see Section 8.3

All the new frames are sent in 1-RTT packets [QUIC-TRANSPORT].

4.1. Path Initiation

Following [QUIC-TRANSPORT], each endpoint uses NEW_CONNECTION_ID frames to issue usable connections IDs to reach it. Before an endpoint adds a new path by initiating path validation, it MUST check whether at least one unused Connection ID is available for each side.

If the transport parameter "active_connection_id_limit" is negotiated as N, the server provided N Connection IDs, and the client is already actively using N paths, the limit is reached. If the client wants to start a new path, it has to retire one of the established paths.

When the multipath option is negotiated, clients that want to use an additional path MUST first initiate the Address Validation procedure with PATH_CHALLENGE and PATH_RESPONSE frames described in Section 8.2 of [QUIC-TRANSPORT]. After receiving packets from the client on a new path, if the server decides to use the new path, the server MUST perform path validation (Section 8.2 of [QUIC-TRANSPORT]) unless it has previously validated that address.

If validation succeed, the client can send non-probing, 1-RTT packets on the new paths. In contrast with the specification in Section 9 of [QUIC-TRANSPORT], the server MUST NOT assume that receiving non-probing packets on a new path indicates an attempt to migrate to that path. Instead, servers SHOULD consider new paths over which non-probing packets have been received as available for transmission.

4.2. Path State Management

An endpoint uses PATH_STATUS frames to inform that the peer should send packets in the preference expressed by these frames. Notice that the endpoint might not follow the peer’s advertisements, but the PATH_STATUS frame is still a clear signal of suggestion for the preference of path usage by the peer.

PATH_STATUS frame describes 2 kinds of path states:

- Mark a path as "available", i.e., allow the peer to use its own logic to split traffic among available paths.

- Mark a path as "standby", i.e., suggest that no traffic should be sent on that path if another path is available.

Endpoints use Destination Connection ID Sequence Number field in PATH_STATUS frame to identify which path state is going to be changed. Notice that PATH_STATUS frame can be sent via a different path. An Endpoint MAY ignore the PATH_STATUS frame if it would make all the paths unavailable in a single connection.

4.3. Path Close

Each endpoint manages the set of paths that are available for transmission. At any time in the connection, each endpoint can decide to abandon one of these paths, following for example changes in local connectivity or changes in local preferences. After an

endpoint abandons a path, the peer will not receive any more non-probing packets on that path. Non-probing packets are defined in Section 9.1 of [QUIC-TRANSPORT].

An endpoint that wants to close a path SHOULD use explicit request to terminate the path by sending the PATH_ABANDON frame (see Section 4.3.1). Note that while abandoning a path will cause Connection ID retirement, only retiring the associated Connection ID does not necessarily advertise path abandon (see Section 4.3.3). However, implicit signals such as idle time or packet losses might be the only way for an endhost to detect path closure (see Section 4.3.4).

Note that other explicit closing mechanisms of [QUIC-TRANSPORT] still apply on the whole connection. In particular, the reception of either a CONNECTION_CLOSE (Section 10.2 of [QUIC-TRANSPORT]) or a Stateless Reset (Section 10.3 of [QUIC-TRANSPORT]) closes the connection.

4.3.1. Use PATH_ABANDON Frame to Close a Path

Both endpoints, namely the client and the server, can initiate path closure, by sending a PATH_ABANDON frame (see Section 8.2) which requests the peer to stop sending packets with the corresponding Destination Connection ID.

The sender and receiver of a PATH_ABANDON frame should not release its resources immediately, but SHOULD wait for at least three times the current Probe Timeout (PTO) interval as defined in Section 6.2. of [QUIC-RECOVERY] after the last sent packet before sending the RETIRE_CONNECTION_ID frame for the corresponding CID. This is inline with the requirement of Section 10.2 of [QUIC-TRANSPORT] to ensure that paths close cleanly and that delayed or reordered packets are properly discarded. The effect of receiving a RETIRE_CONNECTION_ID frame is specified in the next section.

Usually, it is expected that the PATH_ABANDON frame is used by the client to indicate to the server that path conditions have changed such that the path is or will be not usable anymore, e.g. in case of a mobility event. The PATH_ABANDON frame therefore recommends to the receiver that no packets should be sent on that path anymore. In addition, the RETIRE_CONNECTION_ID frame is used indicate to the receiving peer that the sender will not send any packets associated to the Connection ID used on that path anymore. The receiver of a PATH_ABANDON frame MAY also send a PATH_ABANDON frame to indicate its own unwillingness to receive any packet on this path anymore.

PATH_ABANDON frames can be sent on any path, not only the path that is intended to be closed. Thus, a path can be abandoned even if connectivity on that path is already broken.

Retransmittable frames, that have previously been sent on the abandoned path and are considered lost, will be retransmitted on a different path.

If a PATH_ABANDON frame is received for the only active path of a QUIC connection, the receiving peer SHOULD send a CONNECTION_CLOSE frame and enter the closing state. If the client received a

PATH_ABANDON frame for the last open path, it MAY instead try to open a new path, if available, and only initiate connection closure if path validation fails or a CONNECTION_CLOSE frame is received from the server. Similarly the server MAY wait for a short, limited time such as one PTO if a path probing packet is received on a new path before sending the CONNECTION_CLOSE frame.

4.3.2. Refusing a New Path

An endpoint may deny the establishment of a new path initiated by its peer during the address validation procedure. According to [QUIC-TRANSPORT], the standard way to deny the establishment of a path is to not send a PATH_RESPONSE in response to the peer's PATH_CHALLENGE.

4.3.3. Effect of RETIRE_CONNECTION_ID Frame

Receiving a RETIRE_CONNECTION_ID frame causes an endpoint to discard the resources associated with that Connection ID. Specifically, the endpoint should not use the sequence number of the retired Connection ID anymore in any control frames, as the peer will not be able to associate those frames to a path and will therefore ignore them. This means an endpoint is also not required to acknowledge any late packets carrying that Connection ID and, hence, it can remove the list of received packets used to send acknowledgements after receiving the RETIRE_CONNECTION_ID frame.

The peer, that sent RETIRE_CONNECTION_ID frame, can keep sending data using the same IP addresses and UDP ports previously associated with that Connection ID, but MUST use a different connection ID when doing so. If no new connection ID is available anymore, the endpoint cannot send on this path. This can happen if, e.g., the Connection ID issuer requests retirement of a Connection ID using the Retire Prior To field in the NEW_CONNECTION_ID frame but does provide sufficient new CIDs.

Note that even if a peer cannot send on a path anymore because it does not have a valid Connection ID to use, it can still acknowledge packets received on the path, by sending ACK_MP frames on another path, if available. But also note that, as there is no valid CID associated with the path, the other end cannot send multipath control frames that contain the sequence number of a Connection ID, such as PATH_ABANDON or PATH_STATUS.

If the peer cannot send on a path and no data is received on the path, the idle time-out will close the path. If, before the idle timer expires, a new Connection ID gets issued by its peer, the endpoint can re-activate the path by sending a packet with a new Connection ID on that path.

If the sender retires a Connection ID that is still used by in-flight packets, it may receive ACK_MP frames referencing the retired Connection ID. If the sender stops tracking sent packets with retired Connection ID, these would be spuriously marked as lost. To avoid such performance issue without keeping retired Connection ID state, an endpoint should first stop sending packets with the to-be-retired Connection ID, then wait for all in-flight packets to be

either acknowledged or marked as lost, and finally retire the Connection ID.

4.3.4. Idle Timeout

[QUIC-TRANSPORT] allows for closing of connections if they stay idle for too long. The connection idle timeout in multipath QUIC is defined as "no packet received on any path for the duration of the idle timeout". When only one path is available, servers MUST follow the specifications in [QUIC-TRANSPORT].

When more than one path is available, hosts shall monitor the arrival of non-probing packets and the acknowledgements for the packets sent over each path. Hosts SHOULD stop sending traffic on a path if for at least the period of the idle timeout as specified in Section 10.1. of [QUIC-TRANSPORT] (a) no non-probing packet was received or (b) no non-probing packet sent over this path was acknowledged, but MAY ignore that rule if it would disqualify all available paths. To avoid idle timeout of a path, endpoints can send ack-eliciting packets such as packets containing PING frames (Section 19.2 of [QUIC-TRANSPORT]) on that path to keep it alive. Sending periodic PING frames also helps prevent middlebox timeout, as discussed in Section 10.1.2 of [QUIC-TRANSPORT].

Server MAY release the resource associated with paths for which no non-probing packet was received for a sufficiently long path-idle delay, but SHOULD only release resource for the last available path if no traffic is received for the duration of the idle timeout, as specified in Section 10.1 of [QUIC-TRANSPORT]. This means if all paths remain idle for the idle timeout, the connection is implicitly closed.

Server implementations need to select the sub-path idle timeout as a trade- off between keeping resources, such as connection IDs, in use for an excessive time or having to promptly reestablish a path after a spurious estimate of path abandonment by the client.

4.4. Path States

Figure 1 shows the states that an endpoint's path can have.

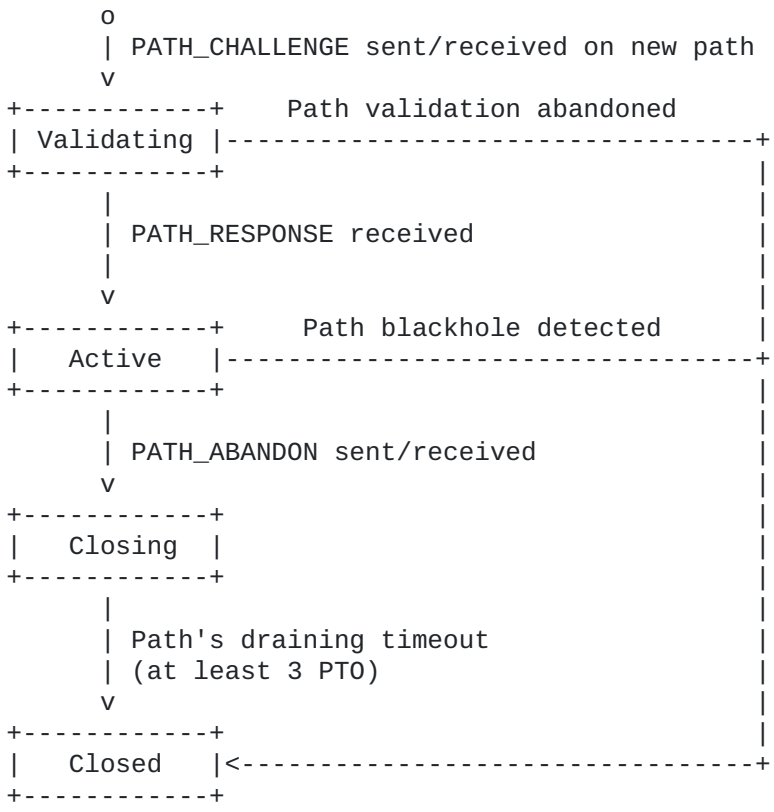

Figure 1: States of a path

In non-final states, hosts have to track the following information.

- Associated 4-tuple: The tuple (source IP, source port, destination IP, destination port) used by the endhost to send packets over the path.

- Associated Destination Connection ID: The Connection ID used to send packets over the path.

In Active state, hosts MUST also track the following information:

- Associated Source Connection ID: The Connection ID used to receive packets over the path. The endpoint relies on its sequence number to send path control information and specifically acknowledge packets belonging to that Connection ID-specific packet number space.

A path in the "Validating" state performs path validation as described in Section 8.2 of [QUIC-TRANSPORT]. An endhost should not send non-probing frames on a path in "Validating" state, as it has no guarantee that packets will actually reach the peer.

The endhost can use all the paths in the "Active" state, provided that the congestion control and flow control currently allow sending of new data on a path. Note that if a path became idle due to a timeout, endpoints SHOULD send PATH_ABANDON frame before closing the path.

In the "Closing" state, the endhost SHOULD NOT send packets on this path anymore, as there is no guarantee that the peer can still map the packets to the connection. The endhost SHOULD wait for the acknowledgment of the PATH_ABANDON frame before moving the path to the "Closed" state to ensure a graceful termination of the path.

When a path reaches the "Closed" state, the endhost releases all the path's associated resources, including the associated Connection IDs. Endpoints SHOULD send RETIRE_CONNECTION_ID frames for releasing the associated Connection IDs following [QUIC-TRANSPORT]. Considering endpoints are not expected to send packets on the current path in the "Closed" state, endpoints can send RETIRE_CONNECTION_ID frames on other available paths. Consequently, the endhost is not able to send nor receive packets on this path anymore.

5. Multipath Operation with Multiple Packet Number Spaces

The QUIC multipath extension uses different packet number spaces for each path. This also means that the same packet number can occur on each path and the packet number is not a unique identifier anymore. This requires changes to the ACK frame as well as packet protection as described in the following subsections.

When multipath is negotiated, each Destination Connection ID is linked to a separate packet number space. Each CID-specific packet number space starts at packet number 0. When following the packet number encoding algorithm described in Appendix A.2 of [QUIC-TRANSPORT], the largest packet number (largest_acked) that has been acknowledged by the peer in this new CID's packet number space is initially set to "None".

5.1. Sending Acknowledgements

The ACK_MP frame, as specified in Section 8.1, is used to acknowledge 1-RTT packets. Compared to the QUIC version 1 ACK frame, the ACK_MP frame additionally contains the receiver's Destination Connection ID Sequence Number field to distinguish the Connection ID-specific packet number space.

Acknowledgements of Initial and Handshake packets MUST be carried using ACK frames, as specified in [QUIC-TRANSPORT]. The ACK frames, as defined in [QUIC-TRANSPORT], do not carry the Destination Connection ID Sequence Number field to identify the packet number space. If the multipath extension has been successfully negotiated, ACK frames in 1-RTT packets acknowledge packets sent with the Connection ID having sequence number 0.

As soon as the negotiation of multipath support is completed, endpoints SHOULD use ACK_MP frames instead of ACK frames to acknowledge application data packets, including 0-RTT packets, using the initial Connection ID with sequence number 0 after the handshake concluded.

ACK_MP frame (defined in Section 8.1) SHOULD be sent on the path it received packet with the Connection ID of the packet number space it acknowledges. However, an ACK_MP frame can be returned via a

different path, based on different strategies of sending ACK_MP frames.

5.2. Packet Protection

Packet protection for QUIC version 1 is specified in Section 5 of [QUIC-TLS]. The general principles of packet protection are not changed for QUIC Multipath. No changes are needed for setting packet protection keys, initial secrets, header protection, use of 0-RTT keys, receiving out-of-order protected packets, receiving protected packets, or retry packet integrity. However, the use of multiple number spaces for 1-RTT packets requires changes in AEAD usage.

Section 5.3 of [QUIC-TLS] specifies AEAD usage, and in particular the use of a nonce, N, formed by combining the packet protection IV with the packet number. When multiple packet number spaces are used, the packet number alone would not guarantee the uniqueness of the nonce.

In order to guarantee the uniqueness of the nonce, the nonce N is calculated by combining the packet protection IV with the packet number and with the Destination Connection ID sequence number.

Section 19 of [QUIC-TRANSPORT] encodes the Connection ID Sequence Number as a variable-length integer, allowing values up to 2^62-1; in this specification, a range of less than 2^32-1 values MUST be used before updating the packet protection key.

To calculate the nonce, a 96 bit path-and-packet-number is composed of the 32 bit Connection ID Sequence Number in byte order, two zero bits, and the 62 bits of the reconstructed QUIC packet number in network byte order. If the IV is larger than 96 bits, the path-and-packet-number is left-padded with zeros to the size of the IV. The exclusive OR of the padded packet number and the IV forms the AEAD nonce.

For example, assuming the IV value is 6b26114b9cba2b63a9e8dd4f, the Connection ID Sequence Number is 3, and the packet number is aead, the nonce will be set to 6b2611489cba2b63a9e873e2.

5.3. Key Update

The Key Phase bit update process for QUIC version 1 is specified in Section 6 of [QUIC-TLS]. The general principles of key update are not changed in this specification. Following QUIC version 1, the Key Phase bit is used to indicate which packet protection keys are used to protect the packet. The Key Phase bit is toggled to signal each subsequent key update. Because of network delays, packets protected with the older key might arrive later than the packets protected with the new key. Therefore, the endpoint needs to retain old packet keys to allow these delayed packets to be processed and it must distinguish between the new key and the old key. In QUIC version 1, this is done using packet numbers so that the rule is made simple: Use the older key if packet number is lower than any packet number frame the current key phase.

When using multiple packet number spaces on different paths, some care is needed when initiating the Key Update process, as different

paths use different packet number spaces but share a single key. When a key update is initiated on one path, packets sent to another path needs to know when the transition is complete. Otherwise, it is possible that the other paths send packets with the old keys, but skip sending any packets in the current key phase and directly jump to sending packet in the next key phase. When that happens, as the endpoint can only retain two sets of packet protection keys with the 1-bit Key Phase bit, the other paths cannot distinguish which key should be used to decode received packets, which results in a key rotation synchronization problem.

To address such a synchronization issue, if key update is initialized on one path, the sender SHOULD send at least one packet with the new key on all active paths. Further, an endpoint MUST NOT initiate a subsequent key update until a packet with the current key has been acknowledged on each path.

Following Section 5.4 of [QUIC-TLS], the Key Phase bit is protected, so sending multiple packets with Key Phase bit flipping at the same time should not cause linkability issue.

6. Examples

6.1. Path Establishment

Figure 2 illustrates an example of new path establishment using multiple packet number spaces.

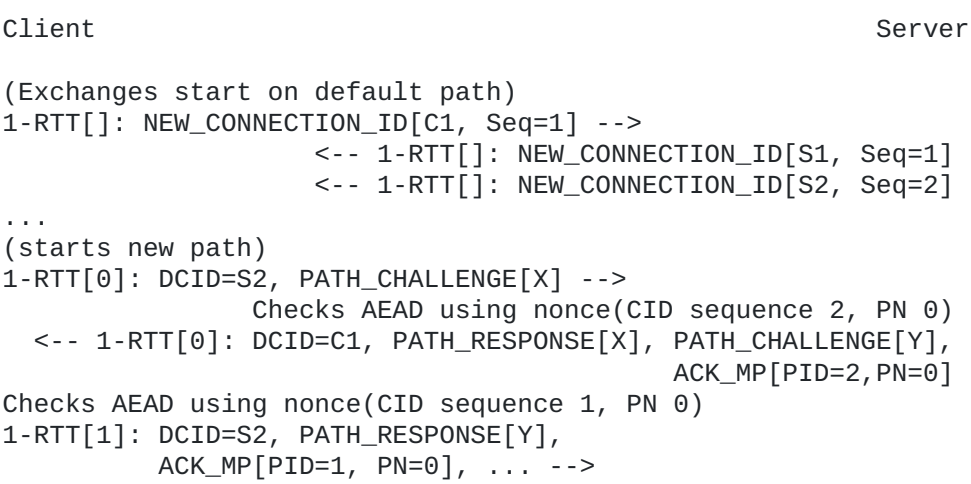

Figure 2: Example of new path establishment

In Figure 2, the endpoints first exchange new available Connection IDs with the NEW_CONNECTION_ID frame. In this example, the client provides one Connection ID (C1 with sequence number 1), and server provides two Connection IDs (S1 with sequence number 1, and S2 with sequence number 2).

Before the client opens a new path by sending a packet on that path with a PATH_CHALLENGE frame, it has to check whether there is an unused Connection IDs available for each side. In this example, the

client chooses the Connection ID S2 as the Destination Connection ID in the new path.

If the client has used all the allocated CID, it is supposed to retire those that are not used anymore, and the server is supposed to provide replacements, as specified in [QUIC-TRANSPORT]. Usually, it is desired to provide one more Connection ID as currently in use, to allow for new paths or migration.

6.2. Path Closure

In this example, the client detects the network environment change (client's 4G/Wi-Fi is turned off, Wi-Fi signal is fading to a threshold, or the quality of RTT or loss rate is becoming worse) and wants to close the initial path.

Figure 3 illustrates an example of path closing. For the first path, the server's 1-RTT packets use DCID C1, which has a sequence number of 1; the client's 1-RTT packets use DCID S2, which has a sequence number of 2. For the second path, the server's 1-RTT packets use DCID C2, which has a sequence number of 2; the client's 1-RTT packets use DCID S3, which has a sequence number of 3. Note that the paths use different packet number spaces. In this case, the client is going to close the first path. It identifies the path by the sequence number of the DCID its peer uses for sending packets over that path, hence using the DCID sequence number 1 (which relates to C1). Optionally, the server confirms the path closure by sending an PATH_ABANDON frame by indicating the sequence number of the DCID the client uses to send over that path, which corresponds to the sequence number 2 (of S2). Both the client and the server can close the path after receiving the RETIRE_CONNECTION_ID frame for that path.

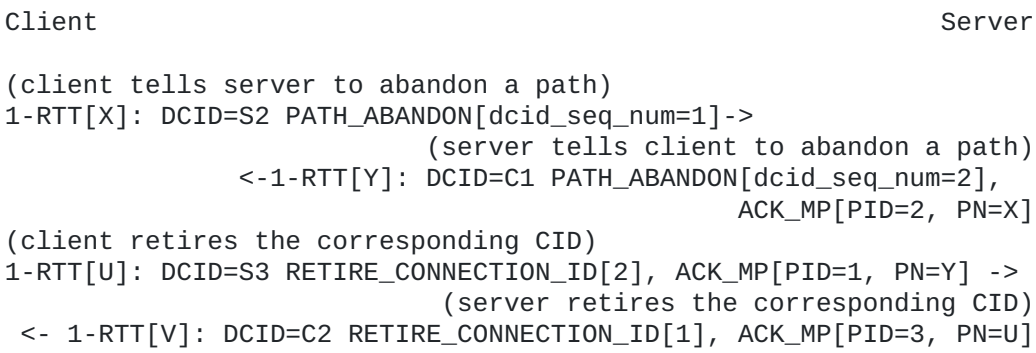

Figure 3: Example of closing a path.

7. Implementation Considerations

7.1. Number Spaces

As stated in Section 1, when multipath is negotiated, each Destination Connection ID is linked to a separate packet number space. This a big difference between implementations of QUIC as specified in [QUIC-TRANSPORT], which only have to manage three number spaces for Initial, Handshake and Application packets.

Implementation of multipath capable QUIC will need to carefully model the relations between paths and number spaces, as shown in Figure 4.

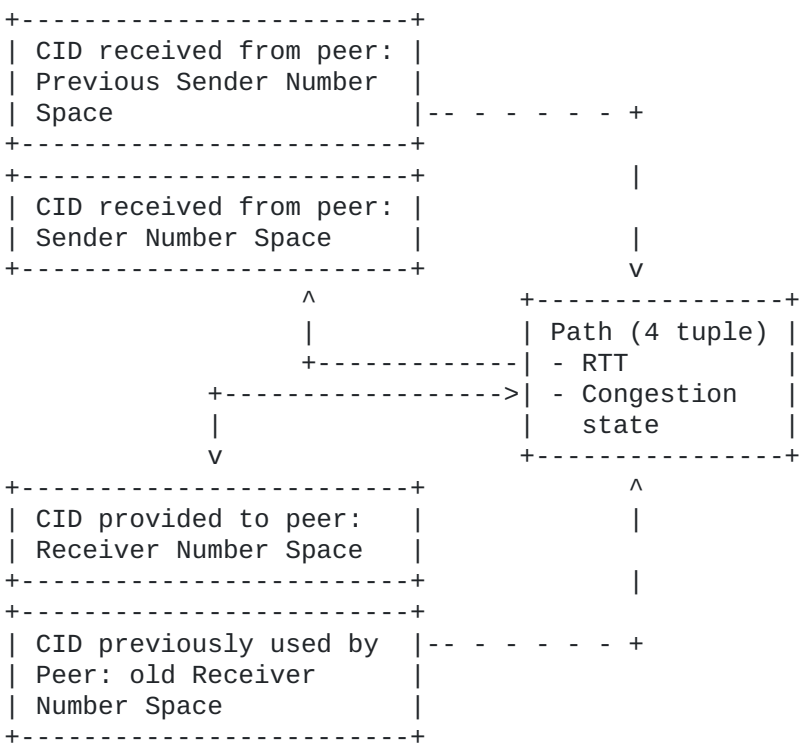

Figure 4: Send and Receive number spaces

The path is defined by the 4-tuple through which packets are received and sent. Packets sent on the path will include the Destination Connection ID currently used for that path, selected from the list of CID provided by the peer. Packets received on the path carry a Destination CID selected by the peer from the list provided to that peer.

The relation between CIDs and paths is not fixed. A node may decide to rotate the Destination CID it uses, a NAT may decide to change the 4-tuple over which packets from that path will be received. Implementation will have to manage these evolving relations.

Data associated with the transmission and reception on a given path can be associated to either the "path state", or to the state of either the sender or receiver number spaces. For example:

- RTT measurements and congestion state are logically associated with the 4-tuple. They will remain unchanged if data starts being received or sent through the same 4-tuple using new CIDs.

- Implementations of loss recovery typically maintain lists of packets sent and not yet acknowledged. Such information, along with the value of the next PN to use for sending, is logically associated with the "Sender Number Space", and with the peer-provided CID used for sending on the path.

*Sending of acknowledgement requires keeping track of the PN of received packets and of acknowledgements previously sent. Such information is logically associated with the "Receiver Number Space", and with the CID used by the peer for sending on the path.

When the link between paths and CID changes, the information tied to the now unused CID remains useful for some time. For example, the list of packet numbers to acknowledge maintained in the old receiver number space could still be used to send ACK_MP frames for that number space. Similarly, the list of packets sent but not yet acknowledged with an old sender number space can be used when processing incoming ACK_MP frames for that number space. Such data should not be discarded immediately after a CID change, but only later, for example when the CID is retired.

7.2. Congestion Control

When the QUIC multipath extension is used, senders manage per-path congestion status as required in Section 9.4 of [QUIC-TRANSPORT]. However, in [QUIC-TRANSPORT] only one active path is assumed and as such the requirement is to reset the congestion control status on path migration. With the multipath extension, multiple paths can be used simultaneously, therefore separate congestion control state is maintained for each path. This means a sender is not allowed to send more data on a given path than congestion control for that path indicates.

When a Multipath QUIC connection uses two or more paths, there is no guarantee that these paths are fully disjoint. When two (or more paths) share the same bottleneck, using a standard congestion control scheme could result in an unfair distribution of the bandwidth with the multipath connection getting more bandwidth than competing single paths connections. Multipath TCP uses the LIA congestion control scheme specified in [RFC6356] to solve this problem. This scheme can immediately be adapted to Multipath QUIC. Other coupled congestion control schemes have been proposed for Multipath TCP such as [OLIA].

7.3. Computing Path RTT

Acknowledgement delays are the sum of two one-way delays, the delay on the packet sending path and the delay on the return path chosen for the acknowledgements. When different paths have different characteristics, this can cause acknowledgement delays to vary widely. Consider for example a multipath transmission using both a terrestrial path, with a latency of 50ms in each direction, and a geostationary satellite path, with a latency of 300ms in both directions. The acknowledgement delay will depend on the combination of paths used for the packet transmission and the ACK transmission, as shown in Table 2.

ACK Path \ Data path	Terrestrial	Satellite
Terrestrial	100ms	350ms
Satellite	350ms	600ms

Table 2: Example of ACK delays using multiple paths

Using the default algorithm specified in [QUIC-RECOVERY] would result in suboptimal performance, computing average RTT and standard deviation from series of different delay measurements of different combined paths. At the same time, early tests showed that it is desirable to send ACKs through the shortest path because a shorter ACK delay results in a tighter control loop and better performances. The tests also showed that it is desirable to send copies of the ACKs on multiple paths, for robustness if a path experiences sudden losses.

An early implementation mitigated the delay variation issue by using time stamps, as specified in [QUIC-Timestamp]. When the timestamps are present, the implementation can estimate the transmission delay on each one-way path, and can then use these one way delays for more efficient implementations of recovery and congestion control algorithms.

If timestamps are not available, implementations could estimate one way delays using statistical techniques. For example, in the example shown in Table 1, implementations can use "same path" measurements to estimate the one way delay of the terrestrial path to about 50ms in each direction, and that of the satellite path to about 300ms. Further measurements can then be used to maintain estimates of one way delay variations, using logical similar to Kalman filters. But statistical processing is error-prone, and using time stamps provides more robust measurements.

7.4. Packet Scheduling

The transmission of QUIC packets on a regular QUIC connection is regulated by the arrival of data from the application and the congestion control scheme. QUIC packets that increase the number of bytes in flight can only be sent when the congestion window allows it. Multipath QUIC implementations also need to include a packet scheduler that decides, among the paths whose congestion window is open, the path over which the next QUIC packet will be sent. Most frames, including control frames (PATH_CHALLENGE and PATH_RESPONSE being the notable exceptions), can be sent and received on any active path. The scheduling is a local decision, based on the preferences of the application and the implementation.

Note that this implies that an endpoint may send and receive ACK_MP frames on a path different from the one that carried the acknowledged packets. A reasonable default consists in sending ACK_MP frames on the path they acknowledge packets, but the receiver must not assume its peer will do so.

7.5. Retransmissions

Simultaneous use of multiple paths enables different retransmission strategies to cope with losses such as: a) retransmitting lost frames over the same path, b) retransmitting lost frames on a different or dedicated path, and c) duplicate lost frames on several paths (not recommended for general purpose use due to the network

overhead). While this document does not preclude a specific strategy, more detailed specification is out of scope.

7.6. Handling different PMTU sizes

An implementation should take care to handle different PMTU sizes across multiple paths. One simple option if the PMTUs are relatively similar is to apply the minimum PMTU of all paths to each path. The benefit of such an approach is to simplify retransmission processing as the content of lost packets initially sent on one path can be sent on another path without further frame scheduling adaptations.

7.7. Keep Alive

The QUIC specification defines an optional keep alive process, see Section 5.3 of [QUIC-TRANSPORT]. Implementations of the multipath extension should map this keep alive process to a number of paths. Some applications may wish to ensure that one path remains active, while others could prefer to have two or more active paths during the connection lifetime. Different applications will likely require different strategies. Once the implementation has decided which paths to keep alive, it can do so by sending Ping frames on each of these paths before the idle timeout expires.

7.8. Connection ID Changes and NAT Rebindings

Section 5.1.2 of [QUIC-TRANSPORT] indicates that an endpoint can change the Connection ID it uses for to another available one at any time during the connection. As such a sole change of the Connection ID without any change in the address does not indicate a path change and the endpoint can keep the same congestion control and RTT measurement state.

While endpoints assign a Connection ID to a specific sending 4-tuple, networks events such as NAT rebinding may make the packet's receiver observe a different 4-tuple. Servers observing a 4-tuple change will performs path validation (see Section 9 of [QUIC-TRANSPORT]). If path validation process succeeds, the endpoints set the path's congestion controller and round-trip time estimator according to Section 9.4 of [QUIC-TRANSPORT].

Section 9.3 of [QUIC-TRANSPORT] allows an endpoint to skip validation of a peer address if that address has been seen recently. However, when the multipath extension is used and an endpoint has multiple addresses that could lead to switching between different paths, it should rather maintain multiple open paths instead.

If an endpoint uses a new Connection ID after an idle period and a NAT rebinding leads to a 4-tuple changes on the same packet, the receiving endpoint may not be able to associate the packet to an existing path and will therefore consider this as a new path. This leads to an inconsistent view of open paths at both peers, however, as the "old" path will not work anymore, it will be silently closed after the idle timeout expires.

8. New Frames

All the new frames MUST only be sent in 1-RTT packet, and MUST NOT use other encryption levels.

If an endpoint receives multipath-specific frames from packets of other encryption levels, it MUST return MP_PROTOCOL_VIOLATION as a connection error and close the connection.

All multipath-specific frames relate to a Destination Connection ID sequence number. If an endpoint receives a Destination Connection ID sequence number greater than any previously sent to the peer, it MUST treat this as a connection error of type MP_PROTOCOL_VIOLATION. If an endpoint receives a multipath-specific frame with a Destination Connection ID sequence number that it cannot process anymore (e.g., because the Connection ID might have been retired), it MUST silently ignore the frame.

8.1. ACK_MP Frame

The ACK_MP frame (types TBD-00 and TBD-01; experiments use 0xbaba00..0xbaba01) is an extension of the ACK frame defined by [QUIC-TRANSPORT]. It is used to acknowledge packets that were sent on different paths using multiple packet number spaces. If the frame type is TBD-01, ACK_MP frames also contain the sum of QUIC packets with associated ECN marks received on the connection up to this point.

ACK_MP frame is formatted as shown in Figure 5.

```
ACK_MP Frame {
  Type (i) = TBD-00..TBD-01 (experiments use 0xbaba00..0xbaba01),
  Destination Connection ID Sequence Number (i),
  Largest Acknowledged (i),
  ACK Delay (i),
  ACK Range Count (i),
  First ACK Range (i),
  ACK Range (..) ...,
  [ECN Counts (..)],
}
```

Figure 5: ACK_MP Frame Format

Compared to the ACK frame specified in [QUIC-TRANSPORT], the following field is added.

Destination Connection ID Sequence Number: The sequence number of the Connection ID identifying the packet number space of the 1-RTT packets which are acknowledged by the ACK_MP frame.

8.2. PATH_ABANDON Frame

The PATH_ABANDON frame informs the peer to abandon a path.

PATH_ABANDON frames are formatted as shown in Figure 6.

```
PATH_ABANDON Frame {
  Type (i) = TBD-02 (experiments use 0xbaba05),
  Destination Connection ID Sequence Number (i),
  Error Code (i),
  Reason Phrase Length (i),
  Reason Phrase (..),
}
```

Figure 6: PATH_ABANDON Frame Format

PATH_ABANDON frames contain the following fields:

Destination Connection ID Sequence Number: The sequence number of the Destination Connection ID used by the receiver of the frame to send packets over the path to abandon.

Error Code: A variable-length integer that indicates the reason for abandoning this path.

Reason Phrase Length: A variable-length integer specifying the length of the reason phrase in bytes. Because an PATH_ABANDON frame cannot be split between packets, any limits on packet size will also limit the space available for a reason phrase.

Reason Phrase: Additional diagnostic information for the closure. This can be zero length if the sender chooses not to give details beyond the Error Code value. This SHOULD be a UTF-8 encoded string [RFC3629], though the frame does not carry information, such as language tags, that would aid comprehension by any entity other than the one that created the text.

PATH_ABANDON frames SHOULD be acknowledged. If a packet containing a PATH_ABANDON frame is considered lost, the peer SHOULD repeat it.

8.3. PATH_STATUS frame

PATH_STATUS Frame are used by endpoints to inform the peer of the current status of one path, and the peer should send packets according to the preference expressed in these frames. PATH_STATUS frames are formatted as shown in Figure 7.

```
PATH_STATUS Frame {
  Type (i) = TBD-03 (experiments use 0xbaba06),
  Destination Connection ID Sequence Number (i),
  Path Status sequence number (i),
  Path Status (i),
}
```

Figure 7: PATH_STATUS Frame Format

PATH_STATUS Frames contain the following fields:

Destination Connection ID Sequence Number:
The sequence number of the Destination Connection ID used by the receiver of this frame to send packets over the path the status update corresponds to.

Path Status sequence number: A variable-length integer specifying the sequence number assigned for this PATH_STATUS frame. The sequence number MUST be monotonically increasing generated by the sender of the PATH_STATUS frame in the same connection. The receiver of the PATH_STATUS frame needs to use and compare the sequence numbers separately for each Destination Connection ID Sequence Number.

Available values of Path Status field are:

*1: Standby

*2: Available

Endpoints use PATH_STATUS frame to inform the peer whether it prefer to use this path or not. If an endpoint receives a PATH_STATUS frame containing 1-Standby status, it SHOULD stop sending non-probing packets on the corresponding path, until it receives a new PATH_STATUS frame containing 2-Available status with a higher sequence number referring to the same path.

Frames may be received out of order. A peer MUST ignore an incoming PATH_STATUS frame if it previously received another PATH_STATUS frame for the same Destination Connection ID Sequence Number with a Path Status sequence number equal to or higher than the Path Status sequence number of the incoming frame.

PATH_STATUS frames SHOULD be acknowledged. If a packet containing a PATH_STATUS frame is considered lost, the peer should only repeat it if it was the last status sent for that path -- as indicated by the sequence number.

9. Error Codes

Multipath QUIC transport error codes are 62-bit unsigned integers following [QUIC-TRANSPORT].

This section lists the defined multipath QUIC transport error codes that can be used in a CONNECTION_CLOSE frame with a type of 0x1c. These errors apply to the entire connection.

MP_PROTOCOL_VIOLATION (experiments use 0xba01): An endpoint detected an error with protocol compliance that was not covered by more specific error codes.

10. IANA Considerations

This document defines a new transport parameter for the negotiation of enable multiple paths for QUIC, and three new frame types. The draft defines provisional values for experiments, but we expect IANA to allocate short values if the draft is approved.

The following entry in Table 3 should be added to the "QUIC Transport Parameters" registry under the "QUIC Protocol" heading.

Value	Parameter Name.	Specification
TBD (current version uses 0x0f739bbc1b666d04)	enable_multipath	Section 3

Table 3: Addition to QUIC Transport Parameters Entries

The following frame types defined in Table 4 should be added to the "QUIC Frame Types" registry under the "QUIC Protocol" heading.

Value	Frame Name	Specification
TBD-00 - TBD-01 (experiments use 0xbaba00-0xbaba01)	ACK_MP	Section 8.1
TBD-02 (experiments use 0xbaba05)	PATH_ABANDON	Section 8.2
TBD-03 (experiments use 0xbaba06)	PATH_STATUS	Section 8.3

Table 4: Addition to QUIC Frame Types Entries

The following transport error code defined in Table 5 should be added to the "QUIC Transport Error Codes" registry under the "QUIC Protocol" heading.

Value	Code	Description	Specification
TBD (experiments use 0xba01)	MP_PROTOCOL_VIOLATION	Multipath protocol violation	Section 9

Table 5: Error Code for Multipath QUIC

11. Security Considerations

TBD

12. Contributors

This document is a collaboration of authors that combines work from three proposals. Further contributors that were also involved one of the original proposals are:

*Qing An

*Zhenyu Li

13. Acknowledgments

TBD

14. References

14.1. Normative References

[QUIC-RECOVERY]

Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <https://www.rfc-editor.org/rfc/rfc9002>.

[QUIC-TLS] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/rfc/rfc9001>.

[QUIC-TRANSPORT] 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>.

[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>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/rfc/rfc3629>.

[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>.

14.2. Informative References

[I-D.bonaventure-iccrg-schedulers] Bonaventure, O., Piraux, M., De Coninck, Q., Baerts, M., Paasch, C., and M. Amend, "Multipath schedulers", Work in Progress, Internet-Draft, draft-bonaventure-iccrg-schedulers-02, 25 October 2021, <https://datatracker.ietf.org/doc/html/draft-bonaventure-iccrg-schedulers-02>.

[OLIA] Khalili, R., Gast, N., Popovic, M., Upadhyay, U., and J. Le Boudec, "MPTCP is not pareto-optimal: performance issues and a possible solution", Proceedings of the 8th international conference on Emerging networking experiments and technologies, ACM , 2012.

[QUIC-Invariants] Thomson, M., "Version-Independent Properties of QUIC", RFC 8999, DOI 10.17487/RFC8999, May 2021, <https://www.rfc-editor.org/rfc/rfc8999>.

[QUIC-Timestamp] Huitema, C., "Quic Timestamps For Measuring One-Way Delays", Work in Progress, Internet-Draft, draft-huitema-quic-ts-08, 28 August 2022, <https://datatracker.ietf.org/doc/html/draft-huitema-quic-ts-08>.

[RFC6356] Raiciu, C., Handley, M., and D. Wischik, "Coupled Congestion Control for Multipath Transport Protocols", RFC 6356, DOI 10.17487/RFC6356, October 2011, <https://www.rfc-editor.org/rfc/rfc6356>.

Authors' Addresses

Yanmei Liu (editor)
Alibaba Inc.

Email: miaoji.lym@alibaba-inc.com

Yunfei Ma
Alibaba Inc.

Email: yunfei.ma@alibaba-inc.com

Quentin De Coninck (editor)
UCLouvain

Email: quentin.deconinck@uclouvain.be

Olivier Bonaventure
UCLouvain and Tessares

Email: olivier.bonaventure@uclouvain.be

Christian Huitema
Private Octopus Inc.

Email: huitema@huitema.net

Mirja Kuehlewind (editor)
Ericsson

Email: mirja.kuehlewind@ericsson.com