Workgroup: QUIC Working Group
Internet-Draft: draft-ietf-quic-multipath-03
Published: 24 October 2022
Intended Status: Standards Track
Expires: 27 April 2023
Authors: Y. Liu, Ed.
Alibaba Inc.
Y. Ma
Alibaba Inc.
Q. De Coninck, Ed.
UCLouvain
O. Bonaventure
UCLouvain and Tessares
C. Huitema
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 27 April 2023.

Copyright Notice

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

- 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.

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].

Some deployments of QUIC use zero-length connection IDs. 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. This extension also specifies a way to use zero-length CID by using the same packet number space on all paths. However, when using the same packet number space on multiple paths, out of order delivery is likely. This causes inflation of the number of acknowledgement ranges and therefore of the the size of ACK frames. Senders that accept to use a single number space on multiple paths when sending to a node using zero-length CID need to take special care to minimize the impact of multipath delivery on loss detection, congestion control, and ECN handling. This proposal specifies algorithms for controlling the size of acknowledgement packets and ECN handling in Section Section 9.1 and Section 9.1.4.

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. 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]. In addition, we define the following terms:

*Path: refers to the 4-tuple {source IP address, source port number, destination IP address, destination port number}. A path refers to "network path" used in [QUIC-TRANSPORT].

*Path Identifier (Path ID): An identifier that is used to identify a path in a QUIC connection at an endpoint. Path Identifier is used in multipath control frames (etc. PATH_ABANDON frame) to identify a path. By default, it is defined as the sequence number of the destination Connection ID used for sending packets on that particular path, but alternative definitions can be used if the length of that connection ID is zero.

*Packet Number Space Identifier (PN Space ID): An identifier that is used to distinguish packet number spaces for different paths. It is used in 1-RTT packets and ACK_MP frames. Each node maintains a list of "Received Packets" for each of the CID that it provided to the peer, which is used for acknowledging packets received with that CID.

The difference between Path Identifier and Packet Number Space Identifier, is that the Path Identifier is used in multipath control frames to identify a path, and the Packet Number Space Identifier is used in 1-RTT packets and ACK_MP frames to distinguish packet number spaces for different paths. Both identifiers have the same value, which is the sequence number of the connection ID, if a non-zero connection ID is used. If the connection ID is zero length, the Packet Number Space Identifier is 0, while the Path Identifier is selected on path establishment.

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 it has been validated. Each endpoint associates a Path identifier to each path. This identifier is notably used when a peer sends a PATH_ABANDON frame to indicate that it has closed the path whose identifier is 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.

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 (TBD - experiments use 0xbabf)

*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 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.

This proposal adds one multipath control frame for path management:

* PATH_ABANDON frame for the receiver side to abandon the path (see Section 12.1)

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

4.1. Path Initiation

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 Path Identifier field in PATH_STATUS frame to identify which path's 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.

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 close a path, by sending PATH_ABANDON frame (see Section 12.1) which abandons the

path with a corresponding Path Identifier. Once a path is marked as "abandoned", it means that the resources related to the path, such as the used connection IDs, can be released. However, information related to data delivered over that path SHOULD not be released immediately as acknowledgments can still be received or other frames that also may trigger retransmission of data on another path.

The endpoint sending the PATH_ABANDON frame SHOULD consider a path as abandoned when the packet that contained the PATH_ABANDON frame is acknowledged. When releasing resources of a path, the endpoint SHOULD send a RETIRE_CONNECTION_ID frame for the connection IDs used on the path, if any.

The receiver of a PATH_ABANDON frame SHOULD NOT release its resources immediately, but SHOULD wait for the reception of the RETIRE_CONNECTION_ID frame for the used connection IDs or 3 RTOs.

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 indicates to the receiving peer that the sender does not intend to send any packets on that path anymore but also recommends to the receiver that no packets should be sent in either direction. The receiver of an PATH_ABANDON frame MAY also send an PATH_ABANDON frame to signal its own willingness to not send any packet on this path anymore.

If connection IDs are used, 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. If no connection IDs are used and the PATH_ABANDON frame has to send on the path that is intended to be closed, it is possible that the packet containing the PATH_ABANDON frame or the packet containing the ACK for the PATH_ABANDON frame cannot be received anymore and the endpoint might need to rely on an idle time out to close the path, as described in Section 4.3.4.

Retransmittable frames, that have previously been sent on the abandoned path and are considered lost, SHOULD 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 enters 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 RTO 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. An endpoint that has negotiated the usage of the multipath extension MAY use an explicit method by sending on another active path a PATH_ABANDON frame containing the Path Identifier of the refused path, but only if the PATH_CHALLENGE arrives in a packet using a non-zero length Connection ID.

4.3.3. Effect of RETIRE_CONNECTION_ID Frame

Receiving a RETIRE_CONNECTION_ID frame causes the endpoint to discard the resources associated with that connection ID. If the connection ID was used by the peer to identify a path from the peer to this endpoint, the resources include the list of received packets used to send acknowledgements. The peer MAY decide to keep sending data using the same IP addresses and UDP ports previously associated with the connection ID, but MUST use a different connection ID when doing so.

Note that 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.

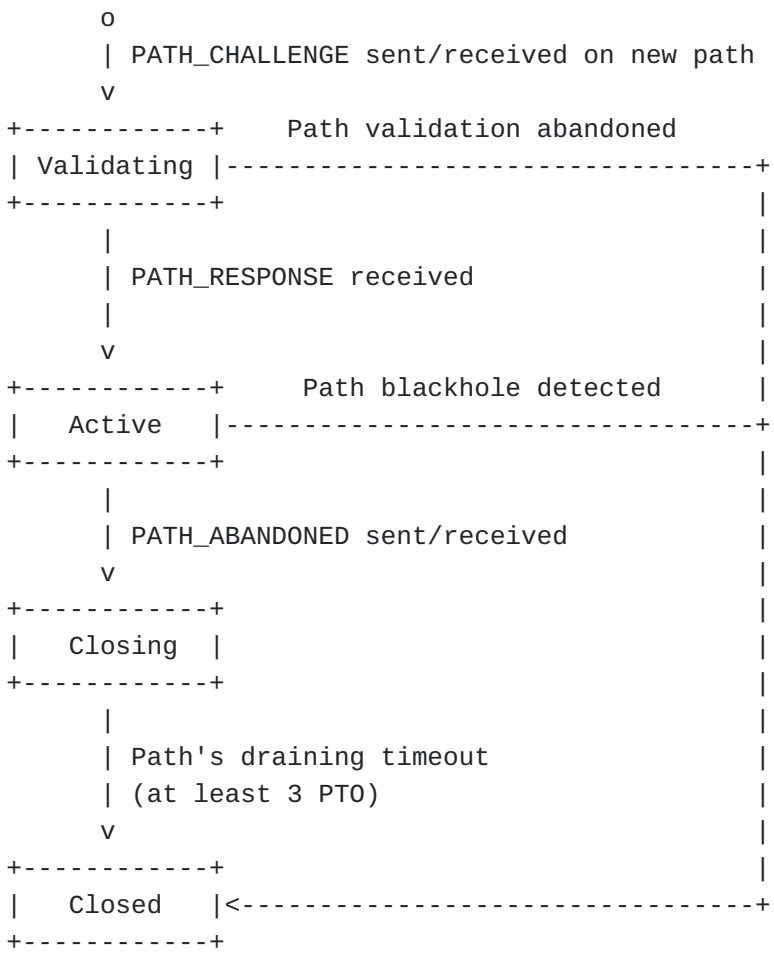

Figure 1: States of a path

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.

If multiple packet number spaces are used over the connection, hosts MUST also track the following information.

- Path Packet Number Space: The endpoint maintains a separate packet number for sending and receiving packets over this path. Packet number considerations described in [QUIC-TRANSPORT] apply within the given path.

In the "Active" state, hosts MUST also track the following information.

- Associated Source Connection ID: The Connection ID used to receive packets over the path.

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_ABANDONED 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_ABANDONED 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. Congestion Control

Senders MUST manage per-path congestion status, and MUST NOT send more data on a given path than congestion control on that path allows. This is already a requirement of [QUIC-TRANSPORT].

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].

6. 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. 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 can only be sent when the congestion window of at least one path is open.

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. Many factors can influence the definition of these algorithms and their precise definition is outside the scope of this document. Various packet schedulers have been proposed and implemented, notably for Multipath TCP. A companion draft [I-D.bonaventure-iccrg-schedulers] provides several general-purpose packet schedulers depending on the application goals.

Note that the receiver could use a different scheduling strategy to send ACK(_MP) frames. The recommended default behaviour consists in sending ACK(_MP) frames on the path they acknowledge packets. Other scheduling strategies, such as sending ACK(_MP) frames on the lowest latency path, might be considered, but they could impact the sender with side effects on, e.g., the RTT estimation or the congestion control scheme. When adopting such asymetrical acknowledgment scheduling, the receiver should at least ensure that the sender negotiated one-way delay calculation mechanism (e.g., [QUIC-Timestamp]).

8. Recovery

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.

9. Packet Number Space and Use of Connection ID

If the connection ID is present (non-zero length) in the packet header, the connection ID is used to identify the path. If no connection ID is present, the 4 tuple identifies the path. The initial path that is used during the handshake (and multipath negotiation) has the path ID 0 and therefore all 0-RTT packets are also tracked and processed with the path ID 0. For 1-RTT packets, the path ID is the sequence number of the Destination Connection ID present in the packet header, as defined in Section 5.1.1 of [QUIC-TRANSPORT], or also 0 if the Connection ID is zero-length.

If non-zero-length Connection IDs are used, an endpoint MUST 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].

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 path identifiers. If for any reason ACK frames are received in 1-RTT packets while the state of multipath negotiation is ambiguous, they MUST be interpreted as acknowledging packets sent on path 0.

9.1. Using Zero-Length connection ID

If a zero-length connection ID is used, one packet number space for all paths. That means the packet sequence numbers are allocated from the common number space, so that, for example, packet number N could be sent on one path and packet number N+1 on another.

In this case, ACK frames report the numbers of packets that have been received so far, regardless of the path on which they have been received. That means the sender needs to maintain an association between sent packet numbers and the path over which these packets were sent. This is necessary to implement per path congestion control, as explained in Section 9.1.2.

Further, the receiver of packets with zero-length connection IDs should implement handling of acknowledgements as defined in Section 9.1.1.

ECN handing is specified in Section 9.1.4, and mitigation of the RTT measurement is further explained in Section 9.1.3.

If a node does not want to implement this logic, it MAY instead limit its use of multiple paths as explained in Section 9.1.5.

9.1.1. Sending Acknowledgements and Handling Ranges

If zero-length CID and therefore also a single packet number space is used by the sender, the receiver MAY send ACK frames instead of ACK_MP frames to reduce overhead as the additional path ID field will anyway always carry the same value.

If senders decide to send packets on paths with different transmission delays, some packets will very likely be received out of order. This will cause the ACK frames to carry multiple ranges of received packets. The large number of range increases the size of ACK frames, causing transmission and processing overhead.

The size and overhead of the ACK frames can be controlled by the combination of one or several of the following:

* Not transmitting again ACK ranges that were present in an ACK frame acknowledged by the peer.

* Delay acknowledgements to allow for arrival of "hole filling" packets.

* Limit the total number of ranges sent in an ACK frame.

* Limiting the number of transmissions of a specific ACK range, on the assumption that a sufficient number of transmissions almost certainly ensures reception by the peer.

* Send multiple messages for a given path in a single socket operation, so that a series of packets sent from a single path uses a series of consecutive sequence numbers without creating holes.

9.1.2. Loss and Congestion Handling With Zero-Length CID

When sending to a zero-length CID receiver, senders may receive acknowledgements that combine packet numbers received over multiple paths. However, even if one packet number space is used on multiple path the sender MUST maintain separate congestion control state for each path. Therefore, senders MUST be able to infer the sending path from the acknowledged packet numbers, for example by remembering which packet was sent on what path. The senders MUST use that information to perform congestion control on the relevant paths, and to correctly estimate the transmission delays on each path. (See Section 9.1.3 for specific considerations about using the ACK Delay field of ACK frames, and Section 9.1.4 for issues on using ECN marks.)

Loss detection as specified in [QUIC-RECOVERY] uses algorithms based on timers and on sequence numbers. When packets are sent over multiple paths, loss detection must be adapted to allow for different RTTs on different paths. When sending to zero-length CID receivers, packets sent on different paths may be received out of order. Therefore, senders cannot directly use the packet sequence numbers to compute the Packet Thresholds defined in Section 6.1.1 of [QUIC-RECOVERY]. Relying only on Time Thresholds produces correct results, but is somewhat suboptimal. Some implementations have been getting good results by not just remembering the path over which a packet was sent, but also maintaining an order list of packets sent on each path. That ordered list can then be used to compute acknowledgement gaps per path in Packet Threshold tests.

9.1.3. RTT Estimation Considerations when SPNS is Used

When SPNS is in use, accurate RTT estimation requires more careful considerations. According to [QUIC-RECOVERY], an endpoint generates an RTT sample on receiving an ACK frame that meets the following two conditions: (1) the largest acknowledged packet number is newly acknowledged, and (2) at least one of the newly acknowledged packets was ack-eliciting. The RTT sample, latest_rtt is calculated as the time elapsed since the largest acknowledged packet was sent. However, when applying the above algorithm with SPNS, one may encounter the following issues: (1) RTT of some paths are not updated timely if ACKs are mostly returned from other paths, and (2) ACK frames depend on the largest received packet of the connection, not the path, and the resulted RTT sample may be the sum of the one-way delays of two different paths. One solution for accurate RTT measurements is to employ time-stamps as described in [QUIC-Timestamp]. If one chooses not to use time-stamps but wants to get reasonable estimation of RTTs on multiple paths with single packet number space, the following practices can be used:

For packet receiver (ACK sender):

- Maintain an ACK threshold and an ACK timer for each path. A path should send an ACK when it receives ack-eliciting-threshold number of ack-eliciting packets (e.g., two) on this path, and an ack-eliciting packet must be acknowledged within MAX_ACK_DELAY.

- Write ACK frame based on the largest received packet of the path. Start the ACK ranges with the largest received packet number of that path, which means that the "Largest Acknowledged" field is the path's largest packet number and the "ACK Delay" field is the delay time of the path's largest received packet.

For packet sender (ACK receiver):

- Maintain an unacked list for each path to retrieve the packets that has been sent when an ACK is received. It can coexist with the unacked list of the connection layer or packet number space layer.

- Generate RTT sample for a path when the following conditions are met: (1) the largest acknowledged packet number is newly acknowledged by the ACK received from this path, and (2) at least one of the newly acknowledged packets was ack-eliciting.

9.1.4. ECN and Zero-Length CID Considerations

ECN feedback in QUIC is provided based on counters in the ACK frame (see Section 19.3.2. of [QUIC-TRANSPORT]). That means if an ACK

frame acknowledges multiple packets, the ECN feedback cannot be accounted to a specific packet.

There are separate counters for each packet number space. However, sending to zero-length CID receivers, the same number space is used for multiple paths. Respectively, if an ACK frames acknowledges multiple packets from different paths, the ECN feedback cannot unambiguously be assigned to a path.

If the sender marks its packets with the ECN capable flags, the network will expect standard reactions to ECN marks, such as slowing down transmission on the sending path. If zero-length CID is used, the sending path is however ambiguous. Therefore, the sender MUST treat a CE marking as a congestion signal on all sending paths that have been by a packet that was acknowledged in the ACK frame signaling the CE counter increase.

A host that is sending over multiple paths to a zero-length CID receiver MAY disable ECN marking and send all subsequent packets as Not-ECN capable.

9.1.5. Restricted Sending to Zero-Length CID Peer

Hosts that are designed to support multipath using multiple number spaces MAY adopt a conservative posture after negotiating multipath support with a peer using zero-length CID. The simplest posture is to only send data on one path at a time, while accepting packets on all acceptable paths. In that case:

* the attribution of packets to path discussed in Section 9.1.2 are easy to solve because packets are sent on a single path,

* the ACK Delays are correct,

* the vast majority of ECN marks relate to the current sending path.

Of course, the hosts will only take limited advantage from the multipath capability in these scenarios. Support for "make before break" migrations will improve, but load sharing between multiple paths will not work.

9.2. Using non-zero length CID and Multiple Packet Number Spaces

If packets contain a non-zero CID, each path has its own packet number space for transmitting 1-RTT packets and a new ACK frame format is used as specified in Section 12.3. Compared to the QUIC version 1 ACK frame, the ACK_MP frames additionally contains a Packet Number Space Identifier (PN Space ID). The PN Space ID used to distinguish packet number spaces for different paths and is

simply derived from the sequence number of Destination Connection ID. Therefore, the packet number space for 1-RTT packets can be identified based on the Destination Connection ID in each packet.

As soon as the negotiation of multipath support is completed, endpoints SHOULD use ACK_MP frames instead of ACK frames for acknowledgements of 1-RTT packets on path 0, as well as for 0-RTT packets that are acknowledged after the handshake concluded.

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.

ACK_MP frame (defined in Section 12.3) can be returned via either a different path, or the same path identified by the Path Identifier, based on different strategies of sending ACK_MP frames.

Using multiple packet number spaces requires changes in the way AEAD is applied for packet protection, as explained in Section 9.2.1, and tighter constraints for key updates, as explained in Section 9.2.2.

9.2.1. Packet Protection for QUIC Multipath

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. If 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 path identifier.

The path ID for 1-RTT packets is the sequence number of [QUIC-TRANSPORT], or zero if the Connection ID is zero-length. 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.

9.2.2. Key Update for QUIC Multipath

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.

10. Examples

10.1. Path Establishment

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

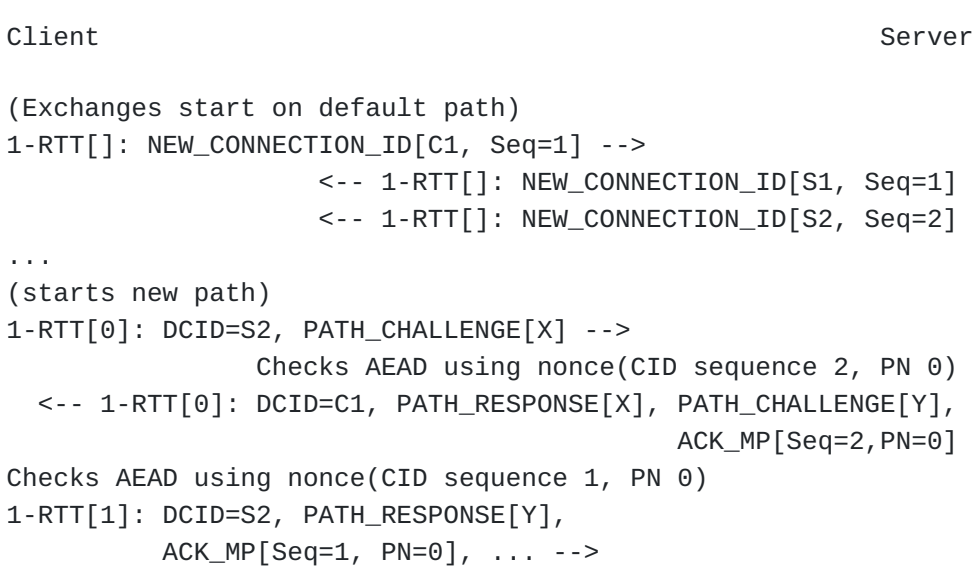

Figure 2: Example of new path establishment

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.

10.2. 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 when both the client and the server use non-zero-length CIDs. 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 received packet's DCID over that path (path identifier type 0x00), hence using the path_id 1. Optionally, the server confirms the path closure by sending an PATH_ABANDON frame using the sequence number of the received packet's DCID over that path (path identifier type 0x00) as path identifier, which corresponds to the path_id 2. Both the client and the server can close the path after receiving the RETIRE_CONNECTION_ID frame for that path.

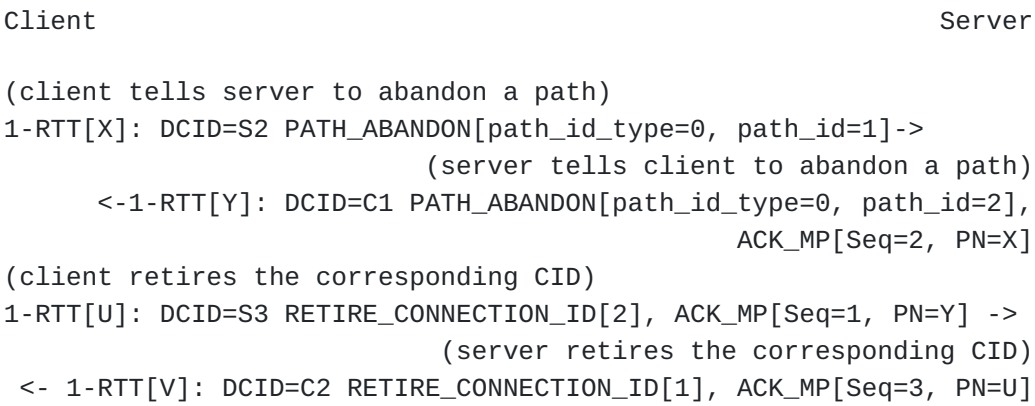

Figure 3: Example of closing a path when both the client and the server choose to receive non-zero-length CIDs.

Figure 4 illustrates an example of path closing when the client chooses to receive zero-length CIDs while the server chooses to receive non-zero-length CIDs. Because there is a zero-length CID in

one direction, single packet number spaces are used. For the first path, the client's 1-RTT packets use DCID S2, which has a sequence number of 2. For the second path, the client's 1-RTT packets use DCID S3, which has a sequence number of 3. Again, in this case, the client is going to close the first path. Because the client now receives zero-length CID packets, it needs to use path identifier type 0x01, which identifies a path by the DCID sequence number of the packets it sends over that path, and hence, it uses a path_id 2 in its PATH_ABANDON frame. The server SHOULD stop sending new data on the path indicated by the PATH_ABANDON frame after receiving it. However, The client may want to repeat the PATH_ABANDON frame if it sees the server continuing to send data. When the client's PATH_ABANDON frame is acknowledged, it sends out a RETIRE_CONNECTION_ID frame for the CID used on the first path. The server can readily close the first path when it receives the RETIRE_CONNECTION_ID frame from the client. However, since the client will not receive a RETIRE_CONNECTION_ID frame, after sending out the RETIRE_CONNECTION_ID frame, the client waits for 3 RTO before closing the path.

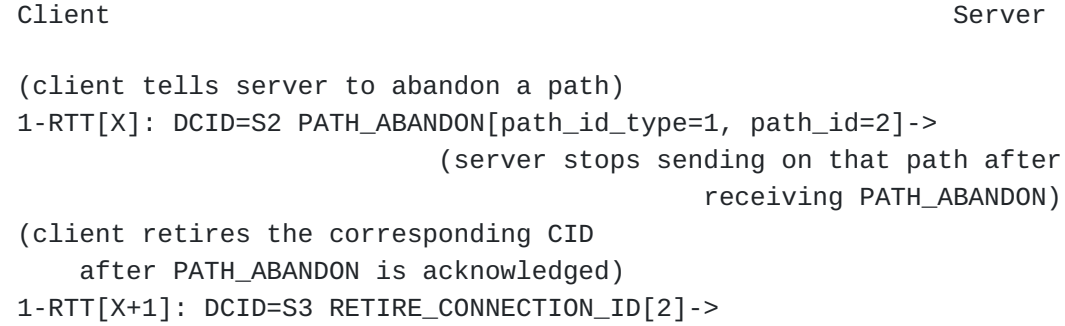

Figure 4: Example of closing a path when the client chooses to receive zero-length CIDs while the server chooses to receive non-zero-length CIDs

11. Implementation Considerations

11.1. 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.

11.2. 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.

12. 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.

12.1. PATH_ABANDON Frame

The PATH_ABANDON frame informs the peer to abandon a path. More complex path management can be made possible with additional extensions (e.g., PATH_STATUS frame in [I-D.liu-multipath-quic]).

PATH_ABANDON frames are formatted as shown in Figure 5.

```
PATH_ABANDON Frame {
  Type (i) = TBD-02 (experiments use 0xbaba05),
  Path Identifier (..),
  Error Code (i),
  Reason Phrase Length (i),
  Reason Phrase (..),
}
```

Figure 5: PATH_ABANDON Frame Format

PATH_ABANDON frames contain the following fields:

Path Identifier: An identifier of the path, which is formatted as shown in Figure 6.

*Identifier Type: Identifier Type field is set to indicate the type of path identifier.

-Type 0: Refer to the connection identifier issued by the sender of the control frame. Note that this is the connection identifier used by the peer when sending packets on the to-be-closed path. This method SHOULD be used if this connection identifier is non-zero length. This method MUST NOT be used if this connection identifier is zero-length.

-Type 1: Refer to the connection identifier issued by the receiver of the control frame. Note that this is the connection identifier used by the sender when sending packets on the to-be-closed path. This method MUST NOT be used if this connection identifier is zero-length.

-Type 2: Refer to the path over which the control frame is sent or received.

*Path Identifier Content: A variable-length integer specifying the path identifier. If Identifier Type is 2, the Path Identifier Content MUST be empty.

```
Path Identifier {
  Identifier Type (i) = 0x00..0x02,
  [Path Identifier Content (i)],
}
```

Figure 6: Path Identifier Format

Note: If the receiver of the PATH_ABANDON frame is using non-zero length Connection ID on that path, endpoint SHOULD use type 0x00 for path identifier in the control frame. If the receiver of the PATH_ABANDON frame is using zero-length Connection ID, but the peer is using non-zero length Connection ID on that path, endpoints SHOULD use type 0x01 for path identifier. If both endpoints are using 0-length Connection IDs on that path, endpoints SHOULD only use type 0x02 for path identifier.

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.

If the Identifier Type is 0x00 or 0x01, PATH_ABANDON frames MAY be sent on any path, not only the path identified by the Path

Identifier Content field. If the Identifier Type if 0x02, the PATH_ABANDON frame MUST only be sent on the path that is intended to be abandoned.

12.2. 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),
  Path Identifier (..),
  Path Status sequence number (i),
  Path Status (i),
}
```

Figure 7: PATH_STATUS Frame Format

PATH_STATUS Frames contain the following fields:

Path Identifier: An identifier of the path, which is formatted as shown in Figure 6. Exactly the same as the definition of Path Identifier in {#path-abandon-frame}.

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 Path Identifier.

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 receive 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 Path Identifier with a sequence number equal to or higher than the 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.

12.3. 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 when 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 8.

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

Figure 8: ACK_MP Frame Format

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

Packet Number Space Identifier: An identifier of the path packet number space, which is the sequence number of Destination Connection ID of the 1-RTT packets which are acknowledged by the ACK_MP frame. If the endpoint receives 1-RTT packets with zero-length Connection ID, it SHOULD use Packet Number Space Identifier 0 in ACK_MP frames. If an endpoint receives an ACK_MP frame with a packet number space ID which was never issued by endpoints (i.e., with a sequence number larger than the largest one advertised), it MUST treat this as a connection error of type MP_PROTOCOL_VIOLATION and close the connection. If an endpoint receives an ACK_MP frame with a packet number space ID which is no more active (e.g., retired by a RETIRE_CONNECTION_ID frame or belonging to closed paths), it MUST ignore the ACK_MP frame without causing a connection error.

When using a single packet number space, endhosts MUST NOT send ACK_MP frames. If an endhost receives an ACK_MP frame while a single packet number space was negotiated, it MUST treat this as a connection error of type MP_PROTOCOL_VIOLATION and close the connection.

13. 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.

14. IANA Considerations

This document defines a new transport parameter for the negotiation of enable multiple paths for QUIC, and two 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 (experiments use 0xbabf)	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 12.3
TBD-02 (experiments use 0xbaba05)	PATH_ABANDON	Section 12.1
TBD-03 (experiments use 0xbaba06)	PATH_STATUS	Section 12.2

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 13

Table 5: Error Code for Multipath QUIC

15. Security Considerations

TBD

16. 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

17. Acknowledgments

TBD

18. References

18.1. Normative References

[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>.

18.2. 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>.

[I-D.liu-multipath-quic] Liu, Y., Ma, Y., Huitema, C., An, Q., and Z. Li, "Multipath Extension for QUIC", Work in Progress, Internet-Draft, draft-liu-multipath-quic-04, 5 September 2021, <https://datatracker.ietf.org/doc/html/draft-liu-multipath-quic-04>.

[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-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-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