QUIC Working Group
Internet-Draft
Intended status: Experimental
Expires: 2 December 2026

W. Gage
Unaffiliated

31 May 2026

QUIC Path Management for Multi-Path Configurations
draft-gage-quic-pathmgmt-06

Abstract

This document defines path management procedures for QUIC that operate independently of the connection management procedures defined in RFC9000. The path management procedures enable a multipath configuration between endpoints by allowing QUIC packets associated with any connection identifier to be transported over any of the paths established between the endpoints. As a consequence, the principles and operations of RFC9000 are retained regardless of the path used to a convey QUIC packet.

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 2 December 2026.

Copyright Notice

Table of Contents

1. Introduction

Architecturally, one may consider two models for data transport over multiple paths: model (A) is a collection of uni-path connection constructs while model (B) is a uni-path connection construct operating over a collection of paths.

Model (A) is like multipath TCP [MPTCP] that uses multiple TCP connections, one for each of the paths. Model (B) is like a single TCP connection operating over a layer 2 link aggregation group [LAG]. In model (B), a TCP segment can be transmitted in an IP datagram over any of the links in the LAG.

In model (B), path management is distinct from connection management. Conceptually, a connection entity sits on top of a path management entity. A packet transmitted by a connection entity is redirected over one of the available paths by the path management entity. A packet received over any of the available paths is redirected by the

path management entity to the connection associated with the packet. The addition, removal and maintenance of paths is handled by the path management entity in a way that is transparent to the connection entities.

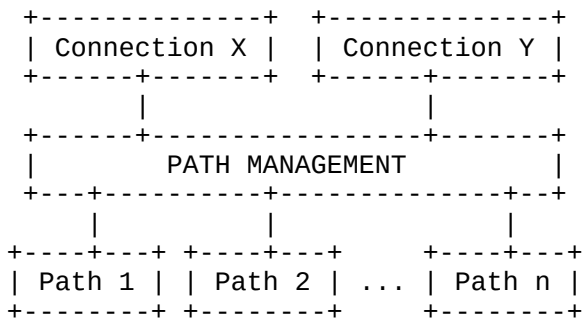

Figure 1: Model (B)

This document describes multi-path QUIC procedures using model (B). In particular, a QUIC packet can be sent over any of the available (and unrestricted) paths. Since connection identifiers are independent of path, a QUIC packet received over any path is processed in the same way as a packet received over the single path construct of [RFC9000] -- i.e. there is a single application data packet number space and an ACK received over any path contains unambiguous packet numbers. While congestion control must clearly be path-specific, connection management, key management and packet loss recovery are not path-specific.

1.1. Conventions

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

1.2. Terminology

This document uses the following terminology:

path: an association with the 4-tuple of an IP/UDP datagram (source IP address, destination IP address, source UDP port, and destination UDP port). The term "path" is used for consistency with other multipath protocols such as [MPTCP] since, in fact, an endpoint has no knowledge of the path a datagram follows through the network beyond the first hop to a network access point.

PathID: path identifier.

PMQUIC: path-managed QUIC (as defined in this document).

session: a collection of QUIC connections and paths used to exchange QUIC packets between two endpoints.

1.3. Notation

This document uses the field notation defined in [RFC9000] and quoted below:

> Individual fields use the following notational conventions, with all lengths in bits:
>
> x (A): Indicates that x is A bits long
>
> x (i): Indicates that x holds an integer value using the variable length encoding described in [RFC9000], Section 16
>
> x (A..B): Indicates that x can be any length from A to B; A can be omitted to indicate a minimum of zero bits, and B can be omitted to indicate no set upper limit; values in this format always end on a byte boundary
>
> x (L) = C: Indicates that x has a fixed value of C; the length of x is described by L, which can use any of the length forms above
>
> x (L) = C..D: Indicates that x has a value in the range from C to D, inclusive, with the length described by L, as above
>
> x (L)...: Indicates that x is repeated zero or more times and that each instance has a length of L

2. Multipath Management

Connection migration to a new path is already supported in [RFC9000]. While [RFC9000] only defines communication over one path at any given time, path-managed QUIC (PMQUIC) provides multiple paths between session endpoints where the paths can be simultaneously active and used to exchange QUIC packets. PMQUIC also provides facilities to explicitly manage the use of paths.

PMQUIC is based on several basic design points:

* Re-use the mechanisms of [RFC9000] as much as possible. In particular, PMQUIC uses path validation based on [RFC9000] and re-uses all of the connection management, key management and loss recovery procedures of [RFC9000].

* Use the same packet header formats as [RFC9000] to avoid differences between multipath and non-multipath traffic over a particular path.

* Do not modify frame formats defined in [RFC9000]; if necessary, define new frame types for path management operations.

PMQUIC changes the path management mechanisms specified in Section 9 of [RFC9000]:

* 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 or lost.

In addition, PMQUIC changes several QUIC path-specific procedures described in [RFC9002]:

* congestion control;

* RTT measurements;

* path maximum payload size discovery.

3. High-level Overview

PMQUIC enables the simultaneous use of different paths to exchange non-probing QUIC frames. This differs from [RFC9000] where the connection migration procedure selects only one path to exchange non-probing frames.

A PMQUIC session between endpoints starts with a standard QUIC handshake over an initial (default) path. As indicated by [RFC9000], an endpoint MUST NOT attempt to activate a new path before the handshake is confirmed. The endpoints use a new max_active_paths transport parameter during the initial cryptographic handshake to negotiate the use of path management capabilities (Section 13.1). The max_active_paths transport parameter indicates support for path management operations and limits the maximum number of active paths that can be used between the endpoints.

To add a new path to an existing PMQUIC session, an endpoint starts a path validation on the chosen path. A new path can only be used to transport non-probing frames once the path has been validated using mechanisms similar to those described in Section 8.2 of [RFC9000]. New PM_CHALLENGE and PM_CHALLENGE_RESPONSE frames are used to validate the path and to assign an identifier to the path. A new PM_STATUS frame may be used to control use of a path and a new PM_ABANDON frame may be used to abandon a path between endpoints, preventing further use of that path to exchange QUIC packets. In addition, a new PM_ADDRESS frame can be used by an endpoint to advertise an IP address that may be used by the peer endpoint to establish a new path.

PM_STATUS and PM_ABANDON frames include a path identifier that is assigned to the affected path, allowing the frame to be forwarded over any of the (allowable) paths active at the time of transmission.

PMQUIC operations do not change the basic operations described in [RFC9000]. In particular, _none_ of the following procedures described in [RFC9000] are affected by the use of multiple paths:

* connection management (e.g. the use of NEW_CONNECTION_ID frames and subsequent rotation of connection identifiers);

* key management (e.g. use of key phase bit) and derivation of AEAD parameters;

* packet loss detection and loss recovery (e.g. using type 0x02 ACK frames).

However, changes to [RFC9002] procedures are required to deal with path-dependent characteristics such as path MTU size, RTT and congestion.

4. Path Identification

A path is associated with the 4-tuple of an IP/UDP datagram (source IP address, destination IP address, source UDP port, and destination UDP port). However, PMQUIC explicitly assigns an identifier to each path to decouple path management from the 4-tuple of the IP/UDP datagram used to transport a QUIC packet.

A path identifier is an integer assigned to a path by an endpoint that unambiguously identifies the path within the session from the perspective of that endpoint. The initial (default) path (i.e. the path used for the exchange of QUIC initial and handshake packets) is implicitly assigned path identifier (PathID) 0 (zero) for the client and PathID 1 (one) for the server. Other than PathID 0 and PathID 1,

each endpoint independently selects the path identifier that it wants to assign to a new path and communicates the chosen PathID to its peer in a PM_CHALLENGE/PM_CHALLENGE_RESPONSE transaction.

An endpoint MUST choose a different PathID for each path in the session -- i.e. a path identifier assigned to one path MUST NOT be reused by the endpoint as the identifier for a different path within the session. For example, a PathID may be a monotonically increasing value, or a randomly generated value, or a sequence of bytes with some internal structure. Since each endpoint independently selects its path identifier, the two endpoints may choose different PathIDs to refer to the same path. A server MAY choose to use the PathID provided by the client in the PM_CHALLENGE frame or the server may choose a different PathID.

A received path identifier that is invalid MUST be treated as a connection error using transport error code Error_pmInvalidPathID (Section 10).

5. Path Activation and Removal

QUIC connections exist and are managed independently of paths. An outgoing QUIC packet may be transmitted over any of the available and active paths, subject to any constraints that may have been placed on path usage by either of the QUIC endpoints (Section 8). Similarly, an incoming QUIC packet received over any path will be processed according to [RFC9000], as though it had been received over a uni-path transport between the QUIC endpoints.

PMQUIC provides mechanisms for adding new paths to a session and for removing unused or unusable paths from a session.

5.1. Path Activation

To initiate communications over a new path, an endpoint MUST send a PM_CHALLENGE frame in the first QUIC packet conveyed over the new path. The PM_CHALLENGE frame contains a new path identifier (PathID) and an unpredictable nonce (Section 11.1).

The PM_CHALLENGE frame is encapsulated (in a QUIC packet) in an IP/UDP datagram where the 4-tuple of the datagram corresponds to the new path. The destination IP address and UDP port may be provided by the peer endpoint (Section 7) or may be discovered by a mechanism that is outside the scope of this document.

To protect against correlation of communications across different IP addresses, it is RECOMMENDED that an endpoint use a new destination connection identifier in the QUIC packet containing the PM_CHALLENGE

frame. The new destination connection identifier would have been previously provided by the peer endpoint in a NEW_CONNECTION_ID frame ([RFC9000], Section 19.15).

The peer endpoint confirms use of the new path by sending a PM_CHALLENGE_RESPONSE frame (Section 11.2) that echoes the received nonce and provides a local PathID as a reference for the path (Section 4). Again, it is RECOMMENDED that the peer endpoint use a new destination connection identifier in the QUIC packet containing the PM_CHALLENGE_RESPONSE frame.

In implementations with decoupling between the path management and connection management entities, the PM_CHALLENGE and PM_CHALLENGE_RESPONSE frames MAY be sent in a QUIC packet using a current connection identifier. An endpoint can disable this behaviour by including a disable_path_migration transport parameter in the initial cryptographic handshake (Section 13.2). An endpoint using a zero-length connection identifier MUST NOT include a disable_path_migration transport parameter in the initial handshake.

The peer endpoint may refuse use of the new path by not sending a PM_CHALLENGE_RESPONSE in response to the PM_CHALLENGE or by sending a PM_CHALLENGE_RESPONSE with a path status parameter (Section 12.3.5) set to Status_NotAvailable.

If the initiating endpoint does not receive a confirming PM_CHALLENGE_RESPONSE frame, it may transmit a new PM_CHALLENGE frame using the same (or a different) IP/UDP 4-tuple but MUST use a new PathID and a different nonce.

To guard against reception of a PM_CHALLENGE frame in an IP/UDP datagram with a spoofed source address, an endpoint receiving a PM_CHALLENGE frame on a new path SHOULD send its own PM_CHALLENGE frame in an IP/UDP datagram that is separate from the IP/UDP datagram used to convey its PM_CHALLENGE_RESPONSE frame.

5.2. Path Migration

If an endpoint determines that a non-probing frame is received (in a QUIC packet) in an IP/UDP datagram where the 4-tuple of the datagram corresponds to a new path, this is considered a passive migration event (e.g. due to a NAT rebinding). Similar to [RFC9000], Section 9.3, the endpoint detecting the new path MUST initiate path validation (Section 5.1) by sending a PM_CHALLENGE frame to the peer endpoint.

The endpoint detecting the new path SHOULD send non-probing frames over a different path, if available (Section 8), until a subsequent PM_CHALLENGE_RESPONSE frame is received. The endpoint MAY send non-probing frames over the new path if no other path is available.

5.3. Path Removal

To terminate communications over an established path, an endpoint sends a PM_ABANDON frame (Section 11.4) containing the PathID of the path to be abandoned. A PM_ABANDON frame may be transmitted over any path that is active (and allowable) at the time of transmission. Abandoning a path has no effect on a QUIC connection.

If the endpoint does not receive an ACK to the QUIC packet containing the PM_ABANDON frame, the PM_ABANDON frame may be retransmitted over the same or a different path.

The reason for abandoning a path may be one of the following (Section 15.5):

Reason_Failing
: indicating that the path is failing (e.g. the path is experiencing excessive transmission errors);

Reason_Lost
: indicating that the path is no longer available to the endpoint;

Reason_NoAck
: indicating that the endpoint failed to received ACKs for QUIC packets transmitted over the path;

Reason_Timeout
: indicating that a idle timer expired with no QUIC packets transmitted or received over the path;

Reason_MaxData
: indicating that the maximum amount of data allowed to be sent on the path has been reached.

Reason_Unspecified
: indicating that the reason is unknown or is otherwise unspecified.

6. Path Maintenance

Once a path between endpoints has been validated, PMQUIC provides mechanisms for defining and updating operational parameters related to the path.

6.1. Path Transmission Status

An endpoint may indicate its initial path transmission status in a PM_CHALLENGE frame (Section 11.1) or in the corresponding PM_CHALLENGE_RESPONSE frame (Section 11.2). By default, the initial path transmission status is Status_Available (Section 6.2).

Subsequently, an initiating endpoint may send a PM_STATUS frame (Section 11.3) to inform its peer endpoint of the desired status of a path (Section 6.2) and, optionally, to update operational parameters associated with the path (Section 12.3).

Each PM_STATUS frame includes a status sequence number that is generated by the initiating endpoint; each endpoint maintains it own status sequence number. The status sequence number MUST be a monotonically increasing value and MUST NOT be used more than once within a session.

If the initiating endpoint does not receive an ACK to the QUIC packet containing the PM_STATUS frame, the PM_STATUS frame may be retransmitted over the same or a different path but MUST include a new status sequence number.

The receiving endpoint MUST ignore an incoming PM_STATUS frame if it previously received another PM_STATUS frame with a status sequence number equal to or higher than the status sequence number of the incoming frame.

If the receiving endpoint does not agree with the status change, the receiving endpoint may send a PM_STATUS frame to inform the initiator of its desired status of the path.

A PM_STATUS frame may be transmitted over any path that is active (and allowable) at the time of transmission.

6.2. Path Status

The status of a path may be set to one of the following:

Status_Available
: indicates that the path may used for transmission of a QUIC packet.

Status_Backup
: indicates that the path should not be used for transmission of a QUIC packet if another path exists in a Status_Available state. This path should only be used if no other path exists in a Status_Available state.

Status_Blocked
: indicates that the initiating endpoint has reached the maximum transmitted data limit imposed by a previously received Parameter_pathMaxData path parameter (Section 12.3.1). The receiving endpoint may increase the maximum data limit (and change the status of the path) using a subsequent PATH_STATUS frame (Section 11.3).

Status_NotAvailable
: indicates that the path should not be used for transmission of a QUIC packet. Unlike an abandoned path (Section 5.3), a path with Status_NotAvailable may be moved to Status_Available or Status_Backup when and if allowed by operational considerations.

6.3. Path Precedence

A path precedence is a variable-length integer value that may be used to distinguish between paths when scheduling the transmission of a QUIC packet:

* in general, a path with a higher precedence value is preferred over a path with a lower precedence value;

* multiple paths may be assigned the same precedence value;

* congestion control may override precedence to allow transmission over a less congested path;

Each endpoint independently determines the precedence of a path and communicates that precedence to its peer (Section 12.3.4). The use of the local and peer path precedence values by an endpoint is beyond the scope of this document.

6.4. Path Congestion Control

Congestion control is applied per path, as described in [RFC9002], Section 7. QUIC packets sent on one path do not affect the congestion state of another path.

An endpoint may include a Parameter_pathInitialCWND (Section 12.2.3) in the list of path parameters in a PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame to provide an initial estimate of the congestion window for the path.

If a Parameter_pathInitialCWND is included in a PM_CHALLENGE_RESPONSE frame, this value must be less than the value (if any) included in the corresponding PM_CHALLENGE frame and takes precedence over the value (if any) included in the PM_CHALLENGE frame.

The mechanism used by an endpoint to determine the congestion window for a path is beyond the scope of this document.

6.5. Path RTT Measurements

Round-Trip Time measurements are performed per path, as described in [RFC9002], Section 5. In general, different paths may exhibit different RTTs.

An endpoint may include a Parameter_pathInitialRTT (Section 12.2.2) in the list of path parameters in a PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame to provide an initial estimate of the path RTT.

If a Parameter_pathInitialRTT is included in a PM_CHALLENGE_RESPONSE frame, this value must be greater than the value (if any) included in the corresponding PM_CHALLENGE frame and takes precedence over the value (if any) included in the PM_CHALLENGE frame.

The mechanism used by an endpoint to determine an initial estimate of the path RTT is beyond the scope of this document.

6.6. Path Maximum UDP Payload Size

By default, the maximum UDP payload size for a path is the max_udp_payload_size transport parameter defined in [RFC9000], Section 18.2.

The maximum UDP payload size for a path can be adjusted by including a Parameter_pathPayloadSize (Section 12.2.1) in the list of path parameters in a PM_CHALLENGE frame (Section 11.1) or in a PM_CHALLENGE_RESPONSE frame (Section 11.2).

If a Parameter_pathPayloadSize is included in a PM_CHALLENGE frame, this value takes precedence over the max_udp_payload_size transport parameter.

If a Parameter_pathPayloadSize is included in a PM_CHALLENGE_RESPONSE frame, this value must be less than the value included in (or defaulted by) the PM_CHALLENGE frame and takes precedence over the value included in (or defaulted by) the PM_CHALLENGE frame.

The mechanism used by an endpoint to determine maximum UDP payload size for a path is beyond the scope of this document. For example, the value may be determined by pre-configuration, by using a Path MTU Discovery (PMTUD) mechanism, or as a property of the endpoint.

6.7. Path Idle Timeout

The path idle time is a time interval during which no packets have been received over the path by an endpoint. The lack of received packets may be due to no transmissions over the path by the peer endpoint or due to packet loss on the path to the receiving endpoint.

An endpoint may assign a maximum path idle time to each path -- i.e. different paths may have the same or different idle timeout values. An endpoint may include a Parameter_pathIdleTimeout (Section 12.3.6) in the list of path parameters in a PM_CHALLENGE, PM_CHALLENGE_RESPONSE or PM_STATUS frame. The mechanism used by an endpoint to determine an idle timeout value for a path is beyond the scope of this document.

The effective idle timeout value used by a receiving endpoint on a path is the minimum of the latest values advertised by each endpoint (or the sole latest advertised value if only one endpoint advertises a value). By default, path idle time monitoring is disabled.

The idle timeout value applies only to the indicated path. A session that migrates to a different path cannot assume that the idle timeout value from an existing path applies to the new path.

Path idle time is monitored independently on each path by each receiving endpoint. If an endpoint determines that the idle time on a path exceeds the idle timeout value for the path, the endpoint SHOULD test the liveness of the path by sending one or more PING or another ack-eliciting frames over the path.

If the ack-eliciting frames are not acknowledged, the endpoint MUST abandon the path (Section 5.3). An idle timeout detected on one path does not affect the state of another path and does not affect the state of the QUIC connection between endpoints.

If the ack-eliciting frame is a non-probing frame, the acknowledgement for the ack-eliciting frame may be received on a path that is different from the path where the liveness test was conducted, depending on how the transmission path for the acknowledgement is selected by the peer endpoint (Section 8). This is illustrated in the example of Figure 2:

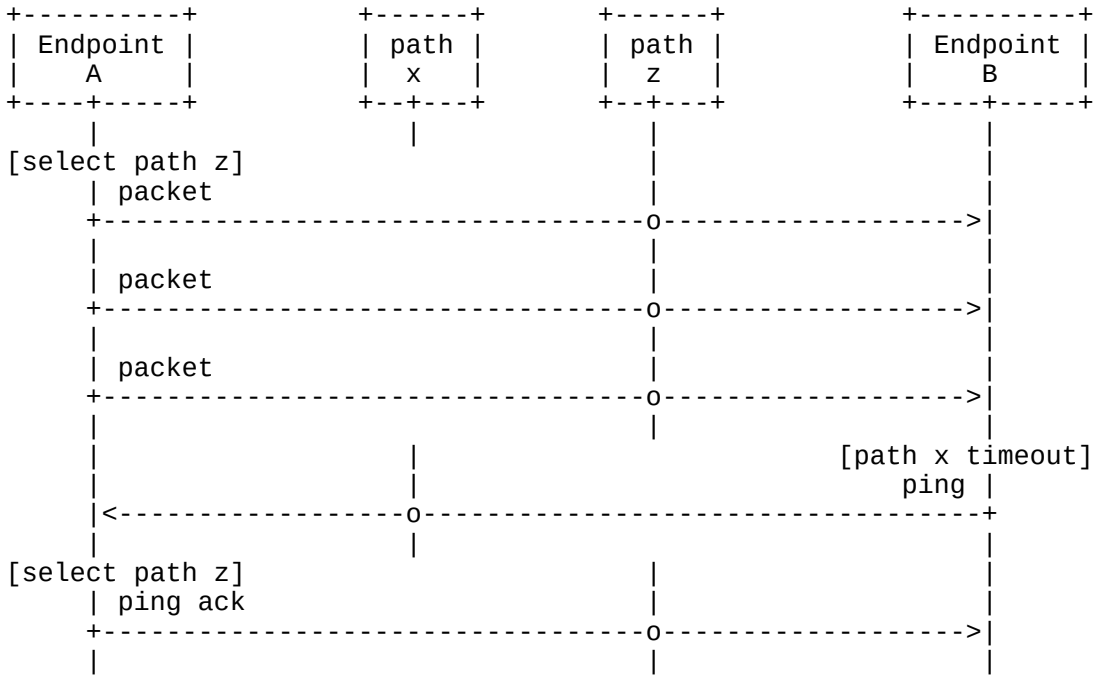

Figure 2: Exemplary Liveness Test

7. Address Advertisement

In [RFC9000], discovery of the IP address used by a peer endpoint is deemed to be beyond the scope of QUIC. PMQUIC, however, provides a mechanism whereby an endpoint can advertise available IP addresses to its peer.

An IP address advertisement can be sent by either a client or a server, allowing establishment of a new path to be initiated by the receiving server or client. For example:

* an IP address advertisement may be used by a multi-homed server to direct client traffic through an alternate IP subnet for load balancing;

* an IP address advertisement may be used by a multi-homed client for improved resiliency;

* an IP address advertisement may be used by a dual-stack endpoint to enable a new path using a different IP address family;

* an IP address advertisement may be used by a receiving endpoint in its route selection process.

An IP address advertisement is conveyed in a PM_ADDRESS frame (Section 11.5). Each PM_ADDRESS frame includes a status sequence number that is generated by the initiating endpoint; the same status sequence number space is used by both PM_ADDRESS and PM_STATUS frames (Section 6.1).

A PM_ADDRESS frame also includes a path status (Section 6.2) indicating the _intended_ status of a path using the advertised address; the intended path status may be superseded by the path status in a subsequent PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame during activation of the resulting path. The intended path status MUST be one of the following:

Status_Available
: indicating that the resulting path may be used for transmission of a QUIC packet;

Status_Backup
: indicating that the resulting path should be used as a backup path (Section 8);

If a previously advertised address is no longer available, the initiating endpoint may revoke the advertised address by sending a PM_ADDRESS frame with the path status set to Status_NotAvailable. If a path has been established using the advertised address, the path MUST be abandoned (Section 5.3) before sending an address advertisement with this status.

A PM_ADDRESS frame may be transmitted over any path that is active (and allowable) at the time of transmission. If the initiating endpoint does not receive an ACK to the QUIC packet containing the PM_ADDRESS frame, the PM_ADDRESS frame may be retransmitted over the same or a different path but MUST include a new status sequence number.

An endpoint receiving a PM_ADDRESS frame MAY use the advertised IP address and UDP port to initiate communications over a new path by following the path activation procedures described in Section 5.1. However an endpoint receiving a PM_ADDRESS frame is not required to make use of the advertised information.

8. Packet Scheduling

A QUIC packet may be scheduled for transmission over a given path only if:

* the path status is either Status_Available or Status_Backup (Section 6.2);

* there is no outstanding PM_ABANDON frame that is pending acknowledgement;

* transmission of the packet does not increase the number of bytes-in-flight beyond the congestion window of the path (Section 6.4);

* transmission of the packet does not cause one of the path limits to be exceeded: path maximum data limit (Section 12.3.1), path maximum bit rate limit (Section 12.3.2), or path maximum packet rate limit (Section 12.3.3).

An endpoint SHOULD schedule a transmission over a path with Status_Available. If this is not possible, the endpoint MAY attempt a transmission over a path with Status_Backup.

If more than one path is eligible for transmission of a packet, the algorithm used to select the path is beyond the scope of this document. An implementation may, for example, use the precedence value provided in a PM_CHALLENGE, PM_CHALLENGE_RESPONSE or PM_STATUS frame (Section 6.3).

Precedence should only used to distinguish between paths with the same status -- i.e. between paths with Status_Available or between paths with Status_Backup.

9. Packet Loss Detection and Recovery

QUIC senders use acknowledgements to detect lost packets and a probe timeout (PTO) to ensure acknowledgements are received. Loss detection through acknowledgements is performed as described in [RFC9002], Section 6.1.

Timer-based loss detection ([RFC9002], Section 6.1.2) must recognise that different paths may exhibit different RTTs (Section 6.5) and SHOULD adjust the packet loss time threshold to accommodate those differences. Probe timeout ([RFC9002], Section 6.2) requires derivation of a PTO period that should also accommodate the different RTT that may be experienced over different paths.

The mechanism used to accommodate those differences in path RTT is beyond the scope of this document.

10. Path Management Connection Errors

This document extends the QUIC Transport Error Codes of [RFC9000], Section 22.5 with the following values (Section 15.6):

Error_pmExceededMaxData
: indicates that the endpoint received more data than allowed over a path (Section 12.3.1).

Error_pmInvalidPathID
: indicates the endpoint received an invalid path identifier (Section 4) -- e.g. a duplicated path identifier, an unknown path identifier, or the identifier of an abandoned path.

Error_pmPathParameter
: indicates that a received path parameter (Section 12) was invalid -- e.g. was badly formatted, included an invalid type, included an invalid value, omitted a mandatory path parameter, included a forbidden path parameter, included a duplicated path parameter, or was otherwise in error.

Error_pmProtocolViolation
: indicates an error with protocol compliance that is not covered by a more specific error code -- e.g. an endpoint received a path management frame when path management is not enabled.

Path management connection errors MUST be processed according to [RFC9000], Section 11.1.

11. Path Management Frame Types

PMQUIC procedures utilise five new QUIC frame types -- PM_CHALLENGE, PM_CHALLENGE_RESPONSE, PM_STATUS, PM_ABANDON and PM_ADDRESS:

* all five path management frame types are ack-eliciting;

* PM_CHALLENGE and PM_CHALLENGE_RESPONSE frames are "probing frames";

* PM_STATUS, PM_ABANDON and PM_ADDRESS are "non-probing frames";

* path management frame types MUST be conveyed in 1-RTT packets and MUST NOT be conveyed in 0-RTT packets.

11.1. PM_CHALLENGE frame

When an endpoint wants to enable use of a new path, it initiates path validation by sending a PM_CHALLENGE frame over the new path. This is analogous to the use of a PATH_CHALLENGE frame in [RFC9000].

A PM_CHALLENGE frame (Figure 3) includes the following fields:

```
PM_CHALLENGE Frame {
  Type (i) = Type_pmChallenge,
  Initiator_PathID (i),
  Nonce (64),
  Path_Parameter_List (8..) ...,
}
```

Figure 3: PM_CHALLENGE Frame Fields

Type
: is set to Type_pmChallenge (Section 15.2).

Initiator_PathID
: is the PathID assigned to the path by the endpoint sending the PM_CHALLENGE frame (Section 4).

Nonce
: is an unpredictable nonce generated by the endpoint for use in this instance of a PM_CHALLENGE frame (Section 5.1).

Path_Parameter_List
: is a list of path parameters (Section 12). Each path parameter in the list of path parameters may be a path configuration group parameter (Section 12.2) or a path operations group parameter (Section 12.3), or the list may be an empty list (Section 12.4). Path parameters not included in the PM_CHALLENGE frame assume their default values.

The QUIC packet containing the PM_CHALLENGE frame MUST include PADDING frames up to the maximum UDP payload size as defined by Parameter_pathPayloadSize (Section 12.2.1), if included in the path parameters, or by the default value if not included in the path parameters.

11.2. PM_CHALLENGE_RESPONSE frame

When an endpoint wants to acknowledge use of a new path, it confirms path validation by sending a PM_CHALLENGE_RESPONSE frame over the new path. This is analogous to the use of a PATH_RESPONSE frame in [RFC9000].

A PM_CHALLENGE_RESPONSE frame (Figure 4) includes the following fields:

```
PM_CHALLENGE_RESPONSE Frame {
  Type (i) = Type_pmChallengeResponse,
  Initiator_PathID (i),
  Responder_PathID (i),
  Nonce (64),
  Path_Parameter_List (8..) ...,
}
```

Figure 4: PM_CHALLENGE_RESPONSE Frame Fields

Type
: is set to Type_pmChallengeResponse (Section 15.2).

Initiator_PathID
: is the PathID included in the corresponding PM_CHALLENGE frame (Section 11.1).

Responder_PathID
: is the PathID assigned to the path by the endpoint sending the PM_CHALLENGE_RESPONSE frame (Section 4).

Nonce
: is the nonce included in the corresponding PM_CHALLENGE frame (Section 11.1).

Path_Parameter_List
: is a list of path parameters (Section 12). Each path parameter in the list of path parameters may be a path configuration group parameter (Section 12.2) or a path operations group parameter (Section 12.3), or the list may be an empty list (Section 12.4). Path parameters not included in the PM_CHALLENGE_RESPONSE frame assume the (default) values indicated by the corresponding PM_CHALLENGE frame.

The QUIC packet containing the PM_CHALLENGE_RESPONSE frame MUST include PADDING frames up to the maximum UDP payload size as defined by Parameter_pathPayloadSize (Section 12.2.1), if specified, or by the default value, if not specified.

11.3. PM_STATUS frame

An endpoint uses a PM_STATUS frame to signal a change in a path parameter.

A PM_STATUS frame (Figure 5) includes the following fields:

```
PM_STATUS Frame {
  Type (i) = Type_pmStatus,
  Receiver_PathID (i),
  Path_Status_Sequence_Number (i),
  Path_Parameter_List (8..) ...,
 }
```

Figure 5: PM_STATUS Frame Fields

Type
: is set to Type_pmStatus (Section 15.2).

Receiver_PathID
: is the PathID assigned to the path by the peer endpoint receiving the PM_STATUS frame.

Path_Status_Sequence_Number
: is the sending endpoint’s sequence number for this PM_STATUS frame (Section 6.1).

Path_Parameter_List
: is a list of path parameters (Section 12). Each path parameter in the list of path parameters MUST be a path operations group parameter (Section 12.3) (i.e. a path parameter MUST NOT be a path configuration group parameter or an empty list parameter).

Note that the status of the path defaults to Status_Available unless explicitly defined by including a Parameter_pathStatus (Section 12.3.5) in the list of path parameters.

11.4. PM_ABANDON frame

An endpoint uses a PM_ABANDON frame to to indicate that it will no longer use the indicated path.

A PM_ABANDON frame (Figure 6) includes the following fields:

```
PM_ABANDON Frame {
  Type (i) = Type_pmAbandon,
  Receiver_PathID (i),
  Reason_Code (i)
}
```

Figure 6: PM_ABANDON Frame Fields

Type
: is set to Type_pmAbandon (Section 15.2).

Receiver_PathID
: is the PathID assigned to the path by the peer endpoint receiving the PM_ABANDON frame.

Reason_Code
: is the reason that the path is being abandoned (Section 5.3).

11.5. PM_ADDRESS frame

An endpoint uses a PM_ADDRESS frame to advertise an IP address and UDP port that may be used to establish a new path to the initiating endpoint.

A PM_ADDRESS frame (Figure 7) includes the following fields:

```
PM_ADDRESS Frame {
  Type (i) = Type_pmAddress,
  Path_Status_Sequence_Number (i),
  Path_Status (i),
  Port_Number (i)
  Path_Address_Family (i),
  Path_Address (32,128),
 }
```

Figure 7: PM_ADDRESS Frame Fields

Type
: is set to Type_pmAddress (Section 15.2).

Path_Status_Sequence_Number
: is the sending endpoint’s path status sequence number for this PM_ADDRESS frame (Section 6.1).

Path_Status
: is the intended status of a path associated with the IP address (Section 7).

Port_Number
: is a non-zero UDP port number.

Path_Address_Family
: indicates the type of IP address, identified by a socket address family value -- i.e. AF_INET (2) for IPv4 or AF_INET6 (10) for IPv6.

Path_Address
: is the binary 4- or 16-byte value of the advertised IP address, in network (big-endian) byte-order.

12. Path Parameters

Each path parameter in a list of path parameters includes the following fields (Figure 8):

```
Path_Parameter {
  End_of_List (1),
  Path_Parameter_ID (7),
  Path_Parameter_Value (i),
}
```

Figure 8: Path Parameter Encoding

End_of_List
: is a boolean value identifying the last parameter in a list of path parameters. A value of 0 (zero) indicates this is the last parameter; a value of 1 (one) indicates that there is at least one more parameter in the list of path parameters.

Path_Parameter_ID
: uniquely identifies the path parameter (Section 15.3).

Path_Parameter_Value
: is a variable-length integer value assigned to the path parameter.

A path parameter, or a list of path parameters, that is malformed or invalid MUST be treated as a connection error using transport error code Error_pmPathParameter (Section 10).

12.1. Path Parameter Groups

Path parameters are organised into three groups -- a path configuration group, a path operations group, and an empty list group:

* the path configuration group (Section 12.2) consists of parameters used to configure a new path;

* the path operations group (Section 12.3) consists of parameters used to modify the operational state of a path;

* the empty list group (Section 12.4) consists of a special parameter used to indicate that there are no path parameters in a parameter list.

The group associated with a path parameter is determined by by the value of the Path_Parameter_ID (Section 15.3).

12.2. Path Configuration Group Parameters

A path configuration group parameter MAY be included in the path parameter list of a PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame but MUST NOT be included in the path parameter list of a PM_STATUS frame.

12.2.1. Parameter_pathPayloadSize

The path payload size parameter is a variable-length integer value that limits the size of UDP payloads that an endpoint believes can be transmitted over the path and/or the endpoint is willing to receive over the path (Section 6.6), expressed as a number of octets. UDP datagrams with payloads larger than this limit are not likely to be received and/or processed by the endpoint.

The default value for this parameter is the max_udp_payload_size transport parameter defined in [RFC9000], Section 18.2.

12.2.2. Parameter_pathInitialRTT

The path initial RTT parameter is a variable-length integer value that is an estimate of the initial RTT for the path, expressed as a number of milliseconds. This value MAY be used as the initial RTT estimate for path congestion control ([RFC9002], Section 5).

The default value for this parameter is the kInitialRtt value defined in [RFC9002], Appendix A.2.

12.2.3. Parameter_pathInitialCWND

The path initial congestion window parameter is a variable-length integer value that is an estimate of the initial congestion window for the path. This value MAY be used by congestion control as the basis for fast start however, as noted in [RFC9000], Section 9.4, "implementations are advised to be cautious when using saved CC parameters on a new path". Further guidance is provided in [RESUME].

There is no default value for this parameter.

12.3. Path Operations Group Parameters

A path operations group parameter MAY be included in the path parameter list of a PM_CHALLENGE, PM_CHALLENGE_RESPONSE or PM_STATUS frame.

12.3.1. Parameter_pathMaxData

The path maximum data parameter is a variable-length integer value that indicates the maximum amount of data that can be sent on the path by the peer endpoint, expressed as a number of octets. The mechanism used by an endpoint to determine this value is beyond the scope of this document.

The maximum data limit applies only in a single direction -- i.e. from the peer endpoint towards the endpoint defining the path maximum data value. Each endpoint may specify a limit, corresponding to a different direction; the specified limits do not need to be the same.

The maximum data limit applies only to the indicated path. A session that migrates to a different path cannot assume that the maximum data limit from an existing path applies to the new path.

By default, the maximum amount of data that can be sent on the path is not limited.

If included in a PM_STATUS frame (Section 11.3), a maximum data value that is less the previous maximum data value associated with the path MUST be treated as as an invalid path parameter.

Receiving an ack-eliciting packet that exceeds the maximum data value previously authorised for a path MUST be treated as a connection error using transport error code Error_pmExceededMaxData (Section 10).

12.3.2. Parameter_pathMaxBitRate

The path maximum bit rate parameter is a variable-length integer value that indicates the maximum bit rate that data can be sent on the path by the peer endpoint, expressed as a number of octets per second. The mechanism used by an endpoint to determine this value is beyond the scope of this document.

The actual bit rate used on a path is determined by congestion control at a transmitting endpoint, however the actual bit rate (as determined by congestion control) MUST NOT exceed the value indicated by the path maximum bit rate parameter.

The maximum bit rate applies only in a single direction -- i.e. from the peer endpoint towards the endpoint defining the path maximum bit rate value. Each endpoint may specify a limit, corresponding to a different direction; the specified limits do not need to be the same.

The maximum bit rate limit applies only to the indicated path. A session that migrates to a different path cannot assume that the maximum bit rate limit from an existing path applies to the new path.

By default, the maximum bit rate that data can be sent on the path is not limited. The maximum bit rate may change over time due to changing network conditions and may be signalled by an endpoint in a PM_STATUS frame (Section 11.3).

12.3.3. Parameter_pathMaxPacketRate

The path maximum packet rate parameter is a variable-length integer value that indicates the maximum rate that packets can be sent on the path by the peer endpoint, expressed as a number of packets per second. The mechanism used by an endpoint to determine this value is beyond the scope of this document.

The actual packet rate used on a path is determined by congestion control pacing at a transmitting endpoint, however the actual packet rate (as determined by congestion control) MUST NOT exceed the value indicated by the path maximum packet rate parameter.

The maximum packet rate applies only in a single direction -- i.e. from the peer endpoint towards the endpoint defining the path maximum packet rate value. Each endpoint may specify a limit, corresponding to a different direction; the specified limits do not need to be the same.

The maximum packet rate limit applies only to the indicated path. A session that migrates to a different path cannot assume that the maximum packet rate limit from an existing path applies to the new path.

By default, the maximum rate that packets can be sent on the path is not limited. The maximum packet rate may change over time due to changing network conditions and may be signalled by an endpoint in a PM_STATUS frame (Section 11.3).

12.3.4. Parameter_pathPrecedence

The path precedence parameter is a variable-length integer value that indicates the precedence of the path to be used in path selection algorithms (Section 6.3).

There is no default value for this parameter. It is RECOMMENDED that precedence values be limited to the range 0..100.

12.3.5. Parameter_pathStatus

The path status parameter is a variable-length integer value that indicates the current status of the path (Section 6.1).

The default value for this parameter is Status_Available.

12.3.6. Parameter_pathIdleTimeout

The path idle timeout parameter is a variable-length integer value that defines the maximum idle time allowed on the path (Section 6.7), expressed as a number of milliseconds. A value of 0 (zero) indicates that path idle time monitoring is disabled for the path.

The default value for this parameter is 0 (zero).

12.4. Parameter_empty

The empty list parameter indicates that there are no entries in the list of path parameters. If specified, the empty list parameter MUST be the only entry in the list of path parameters. The empty list parameter MUST NOT be included if there are other path parameters in a list of path parameters.

The empty list parameter MUST include a Path_Parameter_ID field but MUST NOT include a Path_Parameter_Value field. The End_of_List field MUST be set to 0 (zero).

An empty list parameter MAY be used in the path parameter list of a PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame but MUST NOT be used in the path parameter list of a PM_STATUS frame.

12.5. Path Parameter List Examples

Empty list:

```
0x0     // End_of_List = 0 (true), Path_Parameter_ID = 0x00
```

Single-entry list:

```
0x03    // End_of_List = 0 (true), Path_Parameter_ID = 0x03
0x04    // Path_Parameter_Value = 0x04
```

Multiple-entry list:

```
0x83    // End_of_List = 1 (false), Path_Parameter_ID = 0x03
0x04    // Path_Parameter_Value = 0x04
0x02    // End_of_List = 0 (true), Path_Parameter_ID = 0x02
0x10    // Path_Parameter_Value = 0x10
```

13. Transport Parameters

PMQUIC defines two new transport parameters that may be encoded in the initial cryptographic handshake ([RFC9000], Section 7.4) -- max_active_paths and disable_path_migration. Endpoints MUST NOT remember the value of the PMQUIC transport parameters they received for use in a subsequent connection ([RFC9000], Section 7.4.1).

13.1. max_active_paths

An endpoint signals support for PMQUIC procedures by including a max_active_paths transport parameter in the initial handshake.

max_active_paths (Section 15.1) is an integer value indicating the maximum number of active paths supported by the initiating endpoint. To enable PMQUIC, an endpoint MUST set the maximum number of active paths to a value greater that 1 (one). The maximum number of active paths allowed in the session is the minimum of the exchanged max_active_paths values.

If a max_active_paths transport parameter value is received that is higher than 255 or less than 2, the receiving endpoint MUST close the connection with an error of type TRANSPORT_PARAMETER_ERROR.

To enable use of PMQUIC procedures, both endpoints in a session MUST include a valid max_active_paths transport parameter in the initial handshake. If either of the endpoints does not include the max_active_paths transport parameter, then the endpoints MUST NOT use any of the PMQUIC procedures or frames defined in this document.

13.2. disable_path_migration

An endpoint can prevent use of a connection identifier on more than one path by including a disable_path_migration transport parameter in the initial handshake.

disable_path_migration (Section 15.1) is a zero-length value where presence of the transport parameter indicates migration is disabled.

If migration is disabled, a peer connection identifier is bound to a single path -- i.e. the peer connection identifier may be used as the destination connection identifier in a QUIC packet for transmission over only one path. The bound path is determined by the the first appearance of the peer connection identifier as the destination connection identifier in a QUIC packet.

If migration is not disabled (i.e. the disable_path_migration transport parameter is not included in the initial handshake), a peer connection identifier may be used as the destination connection identifier in a QUIC packet used for transmission over any available path -- i.e. the connection identifier may appear as the destination connection identifier in different QUIC packets on different paths.

Migration is disabled if at least one of the endpoints includes a disable_path_migration transport parameter in the initial cryptographic handshake.

Receiving a disable_path_migration transport parameter without also receiving a max_active_paths transport parameter MUST be treated as a connection error using transport error code Error_pmProtocolViolation (Section 10).

14. Security Considerations

PMQUIC does not change the operating principles of [RFC9000] and, as such, is subject to the same security considerations as [RFC9000], Section 21.

Specific security considerations, associated with the use of path management procedures, include:

Resource usage
: Due to the simultaneous use of multiple paths between session endpoints, PMQUIC may require additional resources in a client and/or in a server. Resource usage associated with paths can be limited by each endpoint through the max_active_paths transport parameter (Section 13).

Data amplification
: The simultaneous use of multiple paths between session endpoints potentially allows for a higher rate of data exchange than might be possible with only a single path between session endpoints. Since PMQUIC uses a path validation mechanism similar to [RFC9000], the anti-amplification limits of [RFC9000], Section 8.2 are also applicable to PMQUIC.

Further, an endpoint can limit the maximum amount of data that can be sent on a particular path through the PMQUIC Parameter_pathMaxData path parameter (Section 12.3.1). This value may be initially set in a PM_CHALLENGE or PM_CHALLENGE_RESPONSE frame and may be subsequently adjusted in a PM_STATUS frame.

Address spoofing
PMQUIC uses a path validation mechanism (Section 5.1) similar to [RFC9000] to prevent address spoofing over a new path by a malicious intermediate node.

Correlation of activity across multiple paths
Due to the possible decoupling of connection management and path management, PMQUIC recommends but does not mandate that different connection identifiers be used on different paths (Section 5.1). As discussed in [RFC9000], Section 9.5, using the same connection identifier on multiple paths would allow a passive observer to correlate activity between those paths. An endpoint can prevent use of a connection identifier on more than one path by including a disable_path_migration transport parameter in the initial cryptographic handshake (Section 13.2).

15. IANA Considerations

If approved, the following _provisional_ entries will be added to the IANA QUIC Protocol Registry [IANA].

| The values in this section of the Internet Draft are
| preliminary, for testing purposes only.

15.1. New QUIC transport parameters

This document defines two new (preliminary) QUIC transport parameters (Section 13):

* max_active_paths (0x21c7948988209ada)

* disable_path_migration (0x33186fc5d0c7cac3)

15.2. New QUIC frame types

This document defines five new (preliminary) QUIC frame types:

* Type_pmChallenge (0x1ae4ea418795ad60) -- Section 11.1

* Type_pmChallengeResponse (0x12c5938576430d3f) -- Section 11.2

* Type_pmStatus (0x06614d6b80a40a24) -- Section 11.3

* Type_pmAbandon (0x2dde9db26610d041) -- Section 11.4

* Type_pmAddress (0x198a357e6fe41403) -- Section 11.5

15.3. PMQUIC path parameters

This document defines ten PMQUIC path parameters:

* Parameter_empty (0x00) -- Section 12.4

* Parameter_pathMaxData (0x01) -- Section 12.3.1

* Parameter_pathPrecedence (0x02) -- Section 12.3.4

* Parameter_pathStatus (0x03) -- Section 12.3.5

* Parameter_pathMaxBitRate (0x04) -- Section 12.3.2

* Parameter_pathMaxPacketRate (0x05) -- Section 12.3.3

* Parameter_pathIdleTimeout (0x06) -- Section 12.3.6

* Parameter_pathPayloadSize (0x40) -- Section 12.2.1

* Parameter_pathInitialRTT (0x41) -- Section 12.2.2

* Parameter_pathInitialCWND (0x42) -- Section 12.2.3

By convention, the identifier for a path operations group parameter is in the range 0x01..0x3f and the identifier for a path configuration group parameter is in the range 0x40..0x7e. The identifier value 0x7f is reserved.

15.4. PMQUIC path status

This document defines four PMQUIC path status values:

* Status_NotAvailable (0x01)

* Status_Blocked (0x02)

* Status_Backup (0x03)

* Status_Available (0x04)

15.5. PMQUIC path abandon reasons

This document defines five PMQUIC path abandon reason codes:

* Reason_Unspecified (0x00)

* Reason_Failing (0x01)

* Reason_Lost (0x02)

* Reason_NoAck (0x03)

* Reason_Timeout (0x04)

* Reason_MaxData (0x05)

15.6. PMQUIC transport error codes

This document extends the QUIC Transport Error Codes of [RFC9000], Section 22.5 with the following (preliminary) values:

* Error_pmExceededMaxData (0x165ee2f99e0d09fa)

* Error_pmInvalidPathID (0x07c0907b4c18170f)

* Error_pmPathParameter (0x0170789441bc48aa)

* Error_pmProtocolViolation (0x297abbb8bb0b3afa)

16. References

16.1. Normative References

[IANA] Internet Assigned Numbers Authority, "QUIC Protocol Registry", <https://www.iana.org/assignments/quic/>.

[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>.

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

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

[RFC9002] 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>.

16.2. Informative References

[LAG] Wikipedia, "Link aggregation", <https://en.wikipedia.org/wiki/Link_aggregation>.

[MPQUIC] Liu, Y., Ma, Y., De Coninck, Q., Bonaventure, O., Huitema, C., and M. K端hlewind, "Managing multiple paths for a QUIC connection", Work in Progress, Internet-Draft, draft-ietf-quic-multipath-21, 17 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-multipath-21>.

[MPTCP] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/rfc/rfc8684>.

[PIRAUX] Piraux, M. and O. Bonaventure, "Additional addresses for QUIC", Work in Progress, Internet-Draft, draft-piraux-quic-additional-addresses-04, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-piraux-quic-additional-addresses-04>.

[RESUME] Kuhn, N., Stephan, E., Fairhurst, G., Secchi, R., and C. Huitema, "Careful Resume: Convergence of Congestion Control from Retained State", RFC 9959, DOI 10.17487/RFC9959, May 2026, <https://www.rfc-editor.org/rfc/rfc9959>.

Appendix A. Comparison to [MPQUIC]

| This section is provided for information only.

A.1. Adherence to RFC9000

[MPQUIC] diverges from the principles of [RFC9000] in a number of areas, as described below.

A.1.1. Connection identifiers

[MPQUIC] binds every connection identifier to a specific path. A path may be associated with multiple connection identifiers but a connection identifier can only be used on a pre-defined path. A change in in the connection identifier used in a QUIC packet header is used to signal an explicit change in the path.

By contrast, [RFC9000] (and PMQUIC) does not associate a connection identifier with a path -- i.e. connection identifiers are independent of paths.

A.1.2. Connection identifier sequence numbers

[MPQUIC] introduces the concept of multiple connection identifier sequence number spaces with a different connection identifier sequence number space for each path. As a consequence, it is possible for different connection identifiers associated with different paths to be assigned the same connection identifier sequence number.

By contrast [RFC9000] (and PMQUIC) define a single connection identifier sequence number space.

A.1.3. Application data packet number spaces

[MPQUIC] introduces the concept of multiple application data (1RTT) packet number spaces with a different application data number space for each path. As a consequence, it is possible for different QUIC packets transmitted over different paths to be assigned the same packet number.

By contrast [RFC9000] (and PMQUIC) define a single application data packet number space, allowing a packet containing non-probing frames to be forwarded over any of the (allowable) paths active at the time of transmission.

A.1.4. AEAD encryption nonce

Due to the use of a different application data number space for each path, it is possible for different QUIC packets transmitted over different paths to be assigned the same packet number. As a consequence, [MPQUIC] changes the AEAD calculation by using the path identifier as part of AEAD encryption nonce.

By contrast [RFC9000] (and PMQUIC) use a single application data packet number space which ensures that different QUIC packets are assigned different packet numbers regardless of the path used to convey a packet.

A.1.5. Connection management frames and procedures

Due to the use of a different application data number space for each path and the use of a different connection identifier sequence number space for each path, MPQUIC endpoints must use multipath-specific frames for packet acknowledgement (PATH_ACK), assignment of new connection identifiers (PATH_NEW_CONNECTION_ID), and retirement of connection identifier (PATH_RETIRE_CONNECTION_ID).

By contrast, [RFC9000] operations are not affected by the use of PMQUIC procedures which obviates the need for multipath-specific connection management procedures and frames.

A.1.6. Zero-length connection identifiers

Because [MPQUIC] uses connection identifiers to identify paths, a zero-length connection identifier cannot be used with multipath operations.

By contrast, PMQUIC does not associate a connection identifier with a path and allows a QUIC packet to be transmitted over any path, including a QUIC packet with a zero-length connection identifier.

A.2. Additional capabilities

PMQUIC provides a number of capabilities that are not available in [MPQUIC], as described below.

A.2.1. Path configuration

Different paths may have different characteristics, however [MPQUIC] provides no mechanism for configuring a path to account for those differences. By contrast, PMQUIC provides an extensible set of path parameters (Section 12) for configuring operations over a path.

A.2.2. Careful session resumption

PMQUIC includes path parameters to enable the careful resumption of a previous session [RESUME].

A.2.3. Address advertisement

The determination of an endpoint IP address is deemed to be beyond the scope of [MPQUIC]. Independently, QUIC extensions have been proposed (e.g. [PIRAUX]) to allow a server to advertise an alternate IP transport address however use of this capability must be negotiated through a separate transport parameter. By contrast, PMQUIC provides the ability for either a server or a client to advertise additional IP transport addresses (Section 7).

A.2.4. Path identification

A path identifier in [MPQUIC] is a monotonically increasing value and the same path identifier is used by both the client and server. By contrast in PMQUIC, the client and server independently choose the identifier to be associated with a path (Section 4) thereby allowing implementation-specific semantics to be incorporated into a path identifier.

A.2.5. Path status

The status of a path in [MPQUIC] is restricted to two values -- available or backup -- with different frame types used to define the different states. PMQUIC includes an extensible path status parameter (Section 6.2) in multiple frame types allowing more fine-grained control of the path state in various circumstances.

A.2.6. Path precedence

PMQUIC allows both the client and server to assign a precedence to a path to aid in selection of a path for a packet transmission (Section 6.3). By contrast, [MPQUIC] provides no capability for aiding in path selection during packet scheduling.

A.2.7. Symmetric operation

[MPQUIC] is an asymmetric protocol where some operations (e.g. path initiation) can only be performed by the client. By contrast, PMQUIC is a symmetric protocol where either client or server can perform an operation.

Author’s Address

Bill Gage
Unaffiliated
Ottawa
Canada
Email: billgage.ietf@gmail.com