Multipath extension for QUIC

draft-ietf-quic-multipath-05

QUIC meeting @ IETF-117 San Francisco

Yanmei Liu, Yunfei Ma, Quentin De Coninck,
Olivier Bonaventure, Christian Huitema, Mirja Kühlewind

Agenda

- Diff from -04 to -05
- Interop reports
- Open issues
- Next steps

Diff from -04 to -05

1. **New transport parameter value for -05**

- Introduce enable_multipath as a zero-length transport parameter
- enable_multipath (current version uses 0x0f739bbc1b666d05)

- enable_multipath transport parameter MUST NOT be remembered / stored with any session tickets which are used when attempting 0-RTT

Diff from -04 to -05

2. Frame types and error code

- Replace Frame type and by random numbers (PR#254)

Frame types for the current version:

- ACK_MP Frame (0x15228c00 - 0x15228c01)
- PATH_ABANDON Frame (0x15228c05)
- PATH_STATUS Frame (0x15228c06)

Error Code for the current version:

- MP_PROTOCOL_VIOLATION (0x1001d76d3ded42f3)
- Use FRAME_ENCODING_ERROR while receives multipath-specific frames in a different packet type except 1-RTT packet

Diff from -04 to -05

3. CID usage in multi-path

- Remove stale statement regarding zero-length CIDs (PR#225)
- Suggest implementations don't use all CIDs unless rebinding is a non-concern (PR #231)
- Remove must check available CIDs on both sides (PR #243)

Diff from -04 to -05

4. Guidance about RTT computation and ACK_MP scheduling

- Implementations can choose between multiple strategies such as:
  - Sending ACK_MP frames on the path they acknowledge packets
  - Sending ACK_MP frames on the shortest path
- Need to compute smoothedRTT and RTT variance per path, using algorithm specified in Section 5.3 of QUIC-RECOVERY
  - as ACK_MP frames might be received through any path
- Some congestion control functions rely on estimates of the minimum RTT
  - Endpoints might remember the path over which the ACK_MP that produced the minimum RTT was received
  - restart the min RTT computation if that path is abandoned

Diff from -04 to -05

5. Editorial guidance:

- 0-RTT packets could be acknowledged by ACK_MP frames (PR#223)
- Clarify PATH_ABANDON and PATH_STATUS are ack-eliciting (PR #230)
- Clarify ECN counts are per-path, not per connection (PR #228)
- Clarify the effect on transport machinery at PATH_ABANDON + 3PTO (PR #234)
- Clarify nonce usage (PR #245)
- Describe situations when all path are stand-by (PR #238)
- Editorial clarification around preferred_address (PR#256)
- Fix token ambiguity issues (PR#260)

Link to the diff: https://author-tools.ietf.org/iddiff?url2=draft-ietf-quic-multipath-05

Diff from -04 to -05

6. Key update

- Change in key update: Switch to using 3xmaxPTO between key updates (PR#257)
- Endpoints SHOULD wait for at least three times **the largest PTO** among all the paths before initiating a new key update after receiving an acknowledgement that confirms receipt of the previous key update.
- This interval is different from that of QUIC version 1 which used three times the PTO of the only one active path.

Link to the diff: https://author-tools.ietf.org/iddiff?url2=draft-ietf-quic-multipath-05

Interop-report: Implementations with multipath support

server → / client ↓	Ericsson-aioquic	xquic	picoquic	Rask	quiche
Ericsson-aioquic	HVDA		HVDA	HVDA	HVDA
xquic	HVDA	HDVACS	HVDA	H	HVDA
picoquic					
Rask	HVDA		HVDA	HVDA	HVDA
quiche	HVDAC	HVDA	HVDAC		HVDACS

Feature	code	details
Handshake	H	The handshake completes with successful negotiation of enable_multipath transport parameter (both ends indicate 0x01)
Path Validation	V	Client sends PATH_CHALLENGE frame to open a new path and server replies with PATH_RESPONSE
Send data	D	Stream data (of one of more streams) is send on all paths; ACK_MP frames are sent and processed
Path Close	C	Client closes a path with PATH_ABANDON frame
Path status	S	Client sends PATH_STATUS frame
Multipath ACK	A	One endpoint sends data and the other endpoints sends ACK (randomly) on all path independent of where data is received

https://github.com/quicwg/multipath/wiki/QUIC-Implementations-with-multipath-support

Open Issues

- PATH_STATUS (#186)
- Error Codes (#253)
- Sending packets/ACK_MP on non-validated paths (#226/#206/#50/PR #239)
- Handling packets with both new DCID and 4-tuple (#188/ PR #198)
- Separate Path IDs from Connection IDs (#214/#169)

Open Issue: PATH_STATUS (#186/PR #242)

PATH_STATUS defines two “status values”, what about others?

2 main possible options:

1. Split frame into PATH_AVAILABLE and PATH_STANDBY (PR #242). Further status frames may be defined using reserved range type numbers

    ? Ok to share the sequence number space across different frame types ?

2. Keep current PATH_STATUS frame with the “status value” field

    ? Defining behavior for unknown/unexpected values ?

Open Issue: Error Codes (#253)

Only MP_PROTOCOL_VIOLATION error defined

Do we need further error codes?

Open Issue: Sending packets/ACK_MP on non-validated paths (#226/#206/#50/PR #239)

How can a receiver acknowledge packets on a not-yet validated path?

- Send ACK_MP on a (different) validated path

This is because the ACK_MP frame is a non-probing frame.

Should we make the ACK_MP frame a probing one?

Open Issue: Handling packets with both new Destination Connection ID and 4-tuple (#188/PR 198)

How to address unintended client migration while it has changed the DCID?

- Path validation currently required to create new paths

Different options:

1. Server guesses to which existing path the new path corresponds
    - May introduce some mismatch
2. Client bundles RETIRE_CONNECTION_ID(old CID)
    - But server cannot acknowledge new packets from old CID
3. Client bundles PATH_ABANDON(old CID, reason=CID RETIRED)
    - But mismatch between abandoning path vs. abandoning CID
4. Client bundles a new CID_ABANDON(old CID)
    - Maybe a too specific/corner-case frame?

Open issue: Separate Path IDs from Connection IDs

(#214/see also closed issue #169)

Currently: one CID = one (QUIC) path with its own packet number space

- When an endpoint rotates the CID over a given 4-tuple path, it actually corresponds to a new (QUIC) path, with new packet number space
- Endpoints can simultaneously use up to active_connection_id_limit paths
- All paths share a same common key
- May have gaps in the CID sequence number space
- Override the connection migration behavior of RFC9000
- Abandon “path” by sending frame corresponding to the CID used on it
  - ? What if another CID is seen in use on a same 4-tuple than an abandoned CID one ?
    - aka. #188

Open issue: Separate Path IDs from Connection IDs

(#214/see also closed issue #169)

Idea: pre-assign CIDs to given Path IDs using MP_NEW_CONNECTION_ID frames (per-path CID space)

- When an endpoint rotates the CID over a given 4-tuple path, it keeps the same (QUIC) path, with the same packet number space
- Endpoints can simultaneously use up to active_path_limit
  - Each path can have up to active_connection_id_limit CIDs
  - Decoupling network paths (4-tuple) from QUIC paths (packet number spaces)
- Paths may (optionally) use different keys
- No more gaps in CID sequence number space**s**
- May keep “connection migration” per-path
  - One active network path per QUIC path, keep probing/non-probing frames
- Can close a Path ID and release all related CIDs

Next steps

- Fix open issues
- Further interop testing
  - As hackathon’s ones were mostly simple cases
- Support from more implementations
- Security considerations
- Try to release stable version in 6 month?