Multipath extension for QUIC

draft-ietf-quic-multipath-02

QUIC meeting @ IETF-114 Philadelphia

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

Updates in -02

New features and modifications:

- A unified proposal for single PN space and multiple PN spaces (was issue #96 and #87, PR #103)
- Add stand-by status for path management with a new frame type: PATH_STATUS frame (was issue #22, PR #117)
- Add clarification about path closure and path initiation denying (PR #133)
- Add some guidance for ACK scheduling (PR #129)

And some editorial modifications such as updates about figure of Path State Machine, etc.. (see diff: https://www.ietf.org/rfcdiff?url2=draft-ietf-quic-multipath-02)

Open Issues

Still open:

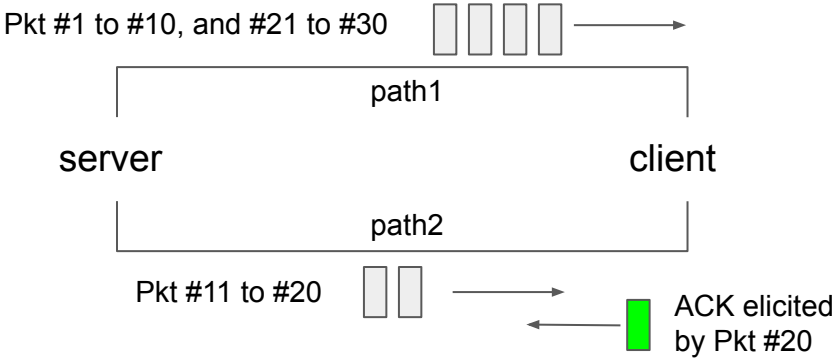
1. Should servers be allowed to open new paths? #47
2. Sending non-probe packets before path validation complete #50
3. Do we need a transport parameter to negotiate max path idle timeout? #95

New but more editorial and both related to delay/RTT calculation

1. Add more explanation on ACK delay and zero-length CID #125 -> has PR #131 but needs more work
2. Can we be more specific about RTT computation? The word "statistical" is a bit broad. #132

Issue #125: Add more explanation on ACK delay and zero-length CID

Considerations of ACK delay and how RTT sample is generated in SPNS



1. Pkt #30 is received on path 1 at t=100ms
2. Pkt #20 is received on path 2 at t=140ms, and an elicited ACK is sent back on path 2 at t=150ms

Q: How to measure RTT on path2 accurately?

Several key points here (more detailed algorithm in PR #131):
1. Need to update path2's RTT even though the largest pkt number(i.e., #30) in the ACK frame is not changed.
2. ACK delay field in ACK frame needs to refer to the receive time of Pkt#20, not Pkt#30.

Issues: 1. change the meaning of the ACK Delay field for a particular use case? 2. RTT sample generation condition is different from RFC9002.

Merged PR #117 (stand-by status for path management)

```
PATH_STATUS Frame {
   Type (i) = TBD-03 (experiments use 0xbaba06),
   Path Identifier (..),
   Path Status sequence number (i),
   Path Status (i),
 }
```

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.

Merged PR #133 (refuse path initiation)

- In RFC9000 establishment of a path is denied by not sending a PATH_RESPONSE
- New:

  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.

Choosing between a single packet number space vs. multiple packet number spaces #96

Quick recap:

	Single PN space	**Multiple PN spaces**
Zero-length CID	Supported	Not supported
Code & ACK Complexity	No new code path if some inefficiency is acceptable. Requires substantial additional code (client & server sides) to manage the ACK size and loss recovery efficiently.	Multiple instantiations of the loss recovery algorithm for each path New ACK Frame keeps small-sized ACK for each path; ACK-Delay and ACK-ECN work as expected with changes needed
Hardware Offload (issue #25)	Out-of order complicates offloading	Change to crypto algo require to include CID

Merged PR: First shot at a unified proposal #103

- **Mandatory** support of multiple packet number spaces (and ACK_MP frame), if multipath extension is negotiated
- Support for zero-length CID (at sender-side) is **optional**
  - Receivers of packets without CID (zero-length) simply set the PN space ID to 0 in ACK_MP
  - Senders that **use multiple paths and send packets without CID** have to implement additional logic for
    - Loss and congestion handling (section 9.1.2)
    - ACK delay calculation (section 9.1.3)
    - ECN handling (section 9.1.4)
  - Or can only only send data on one path at a time

Client SCID	Server SCID	What
long	long	Multiple number spaces
NULL	long	Multiple number spaces on client side (one per CID), single space on server side
long	NULL	Multiple number spaces on server side (one per CID), single space on client side
NULL	NULL	Single number space on each side

Next step

- Editorial pass
  - Specifically on reordering multiple and single PN specific text
- Finalize discussion on remaining open issues
  - Mostly editorial guidance on delay/RTT calculation guidance for single PN space logic
  - Decide if issues #47 (server to open path), #50 (0-RTT-like for new paths) and #95 (new para for path idle timeout) should be addressed now or as potential extension
- More implementation experience is still desired
  - Are people interested in implementing the single PN space logic or not?
  - Do we need zero-length CID support at all?
  - Any additional considerations on hardware offloading?

Backup

Old slides on unsolved issues

Should servers be allowed to open new paths? #47

- In RFC9000 path migration is restricted to clients only (mainly because of problems with NATs).
- However, with multipath, failure of opening a new path is less critical as the old path(s) is not abandoned at the same time.

- Can we release this restriction in the multipath extension?
  - This would support additional use cases.
- Discussed at IETF-113:
  - Action item to review reasoning for restrictions in RFC9000 -> not done yet
  - Could also be a future extension

Sending non-probe packets before path validation complete #50

- Path validation brings at least one round-trip-time delay for data to be sent on the new path
- Can we add a mechanism similar to 0-RTT transmission that still avoids amplification attacks?
- Discussed at IETF-113 but use case was unclear

Do we need a transport parameter to negotiate max path idle timeout? #95

- Currently `max_idle_timeout` is also used for each path to close on idle time-out.
- Do we need to signal separate time-out values per path?
- Discussion so far:
  - Yes, "path idle timeout" might be a way to have a stronger guarantee to stop using paths (closing them)
  - No, just makes the protocol more complex; just use a shorter time-out locally (if it's only one of multiple paths)