

# Multipath Extensions for QUIC

---

Quentin De Coninck (UCLouvain, Belgium)

<https://gdeconinck.github.io/>

5-6 February 2020

QUIC Interim, Zurich

# Why Using Multiple Paths?

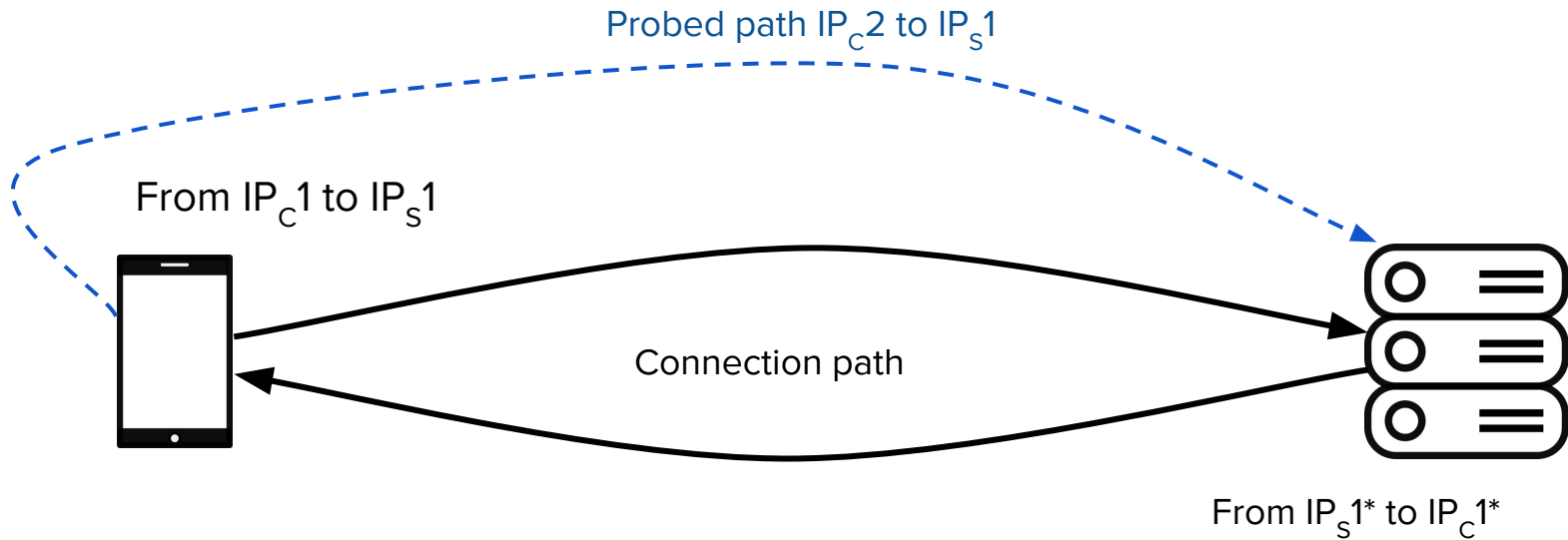
- Bandwidth aggregation
  - MPTCP: Hybrid networks, 4G/WiFi aggregation in Korea
- Network resiliency
  - E.g., duplicate some frames over multiple networks when paths experience connectivity issues
  - MPTCP: iOS deployment
- 3GPP discussions to integrate MPQUIC solution for 5G

# Design Goals

Enable **simultaneous** usage of multiple network paths while

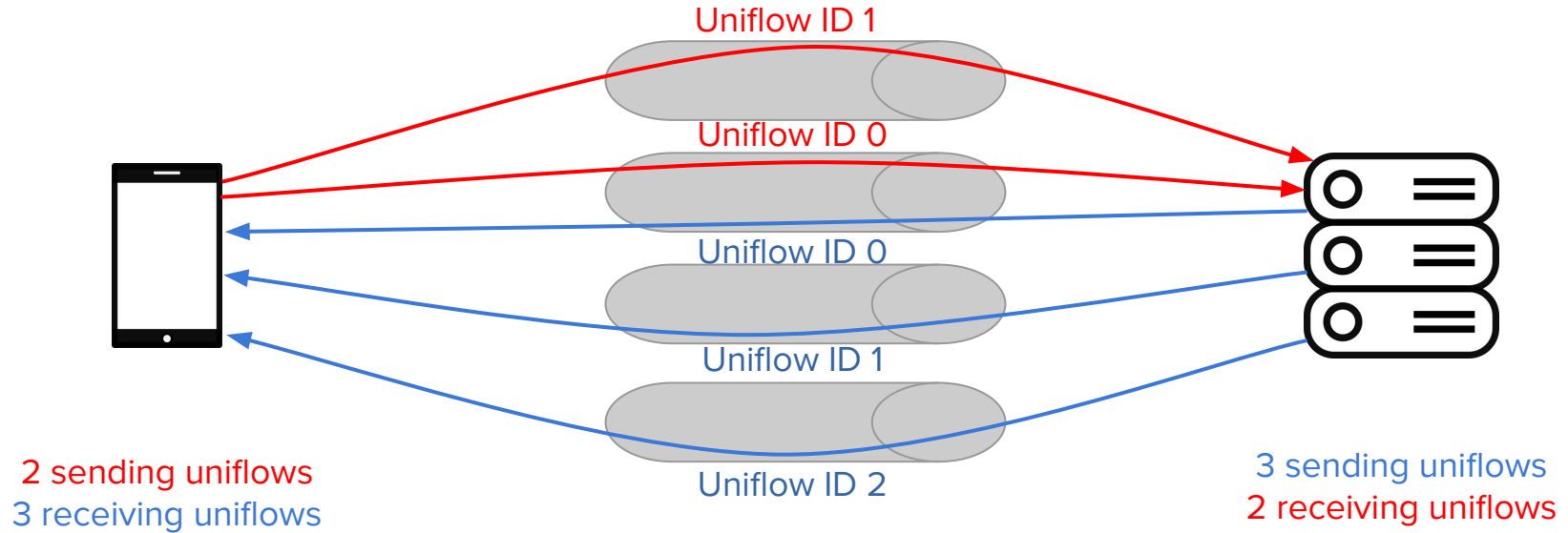
- Handling network asymmetry
- Keeping control on path management
- Not introducing new privacy concerns
- Validating new addresses before use

# Notions of Paths in QUIC



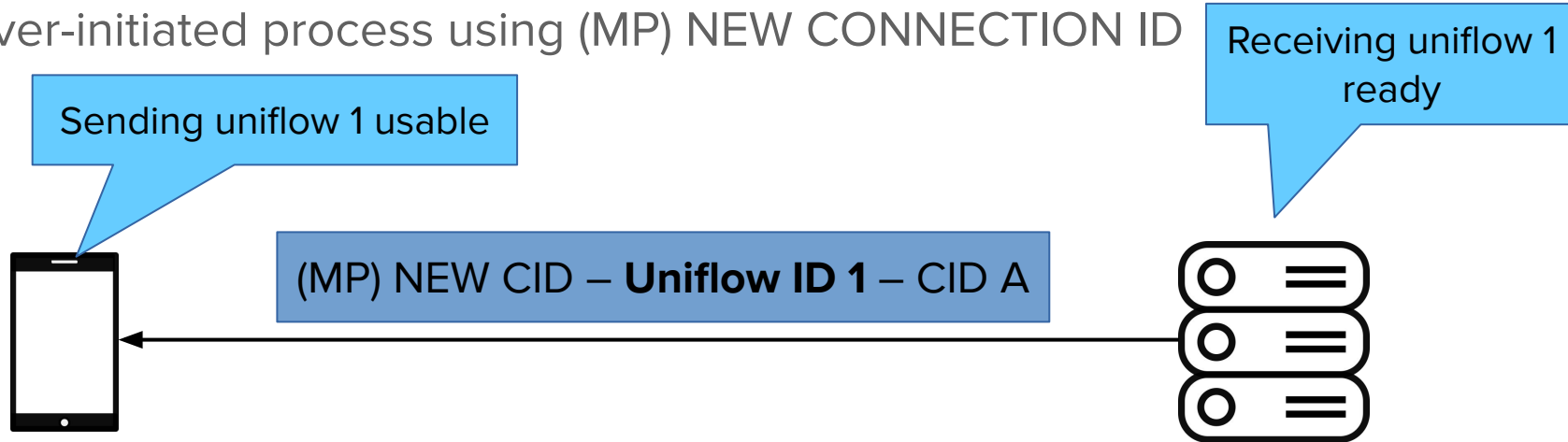
Flow tuples on client and server are not necessarily the same (e.g., NAT)

# Idea of “Asymmetric Flows” (a.k.a. Uniflows)



# Proposing Uniflows

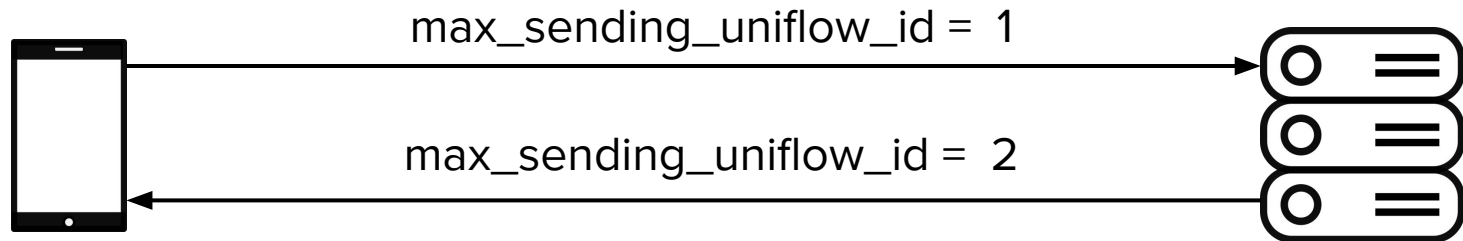
Receiver-initiated process using (MP) NEW CONNECTION ID



- Host determines the number of sending uniflows of its peer
  - = number of different Uniflow IDs sent in (MP) NEW CID frames
- Can only use Destination CID from different Uniflow IDs simultaneously
- Path migration of a given Uniflow still possible

# Negotiating the Multipath Extensions

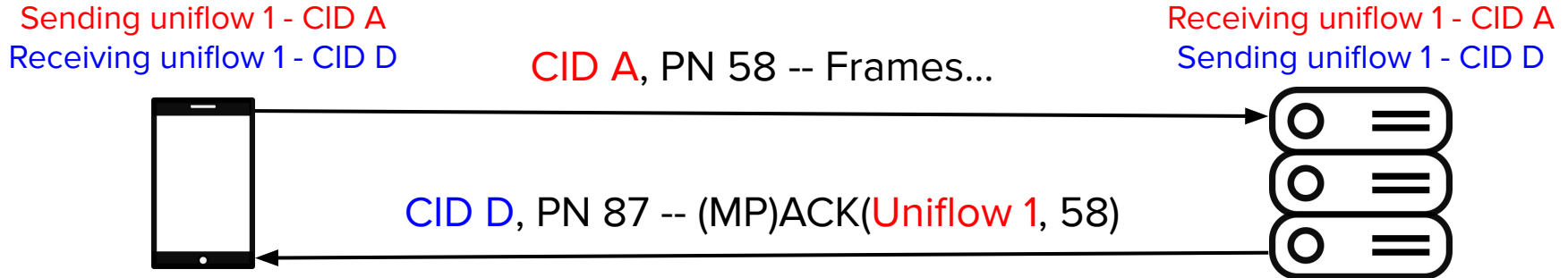
Use a transport parameter



- Put a limit on the number of desired sending uniflows
  - Limit number of peer's receiving uniflows
- Initial path has (implicit) uniflow ID 0
- If one does not advertise the TP, extension is disabled

# Acknowledging Packets Sent on Uniflows

- One packet number space per unifold
- (MP)ACK frame indicates the **receiving Unifold ID** it acknowledges



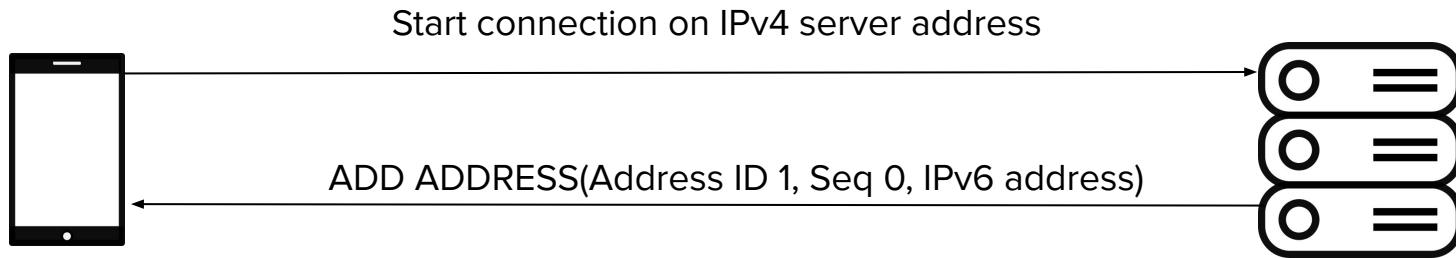
Cons of using a single packet number space:

- May complexify ACK processing, more ACK blocks if path reordering
- What if a single ACK acknowledges two packets from different uniflows?
  - Unifold latency estimation?



# Advertising Host Addresses

Relies on ADD ADDRESS and REMOVE ADDRESS frames



- Similar to preferred\_address here, but can use both IPv4 and IPv6 at the same time
- Enable each host to advertise address changes
- Address ID events ordered by a sequence number

ADD ADDRESS(1, 0, IP1)

ADD ADDRESS(1, 2, IP2)

REMOVE ADDRESS(1, 1)

# Estimating the Latency

(MP)ACK frames can be sent on any (sending) uniflow

- Estimate RTT between tuple (sending uniflow ID, receive uniflow ID)
- Would be interesting to estimate One-Way-Delay of uniflows
  - draft-huitema-quick-1wd

# Multipath Algorithms

- Path management (link unflow to a 4-tuple)
  - Client can link a unflow to pair of local address / remote validated address
  - Server should first wait for client packet to learn client addresses
    - Then link unflows to pair of local address / seen and validated client address
- Packet scheduling
  - Only applies on sending unflows
  - Can be lowest-latency oriented, priority based,...
  - Frames like MAX DATA should be duplicated on all sending unflows
    - To limit head-of-line blocking

# Is This Complex to Implement?

- quickly -- ~350 lines of diff
  - Very basic multipath support with round-robin
- PQUIC multipath plugin (based on picoquic) -- 2500 lines of C code
  - Nearly complete implementation, RTT + round-robin schedulers
  - See <https://pquic.org>
- Connection migration already implements a lot
  - Path validation,...

# Next Steps

Multipath ID is ongoing work, feel free to contribute

- <https://github.com/qdeconinck/draft-deconinck-multipath-quic>

Also preparing a new version of the MultipathTester iOS app

- Include support for IETF (multipath) QUIC
- <https://apps.apple.com/us/app/multipathtester/id1351286809>
- See also <https://multipath-quic.org>