QUIC und HTTP/2 — neue Internet Protokolle

Mirja Kühlewind <mirja.kuehlewind@tik.ee.ethz.ch>

IT Monitoring Konferenz

28. September 2017

mami
measurement and architecture for a middleboxed internet

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688421.The opinions expressed and arguments employed reflect only the authors' view. The European Commission is not responsible for any use that may be made of that information.

Supported by the Swiss State Secretariat for Education, Research and Innovation under contract number 15.0268. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the Swiss Government.

Short Bio — Mirja Kühlewind

- Senior researcher at **Eidgenössische Technische Hochschule (ETH) Zürich** since 2014
  - PhD on TCP congestion control at University of Stuttgart in 2015
  - Dipl.-Ing. in Informations- und Kommunikationstechnik at Friedrich-Alexander-University Erlangen-Nürnberg in 2009
- **Internet Engineering Task Force (IETF)** Area Director for the transport area since 2015
  - Co-chair of the Internet Research Task Force (IRTF) Measurement and Analysis for Protocols Research Group (MAPRG)
  - Previously (till March 2015) co-chair IETF rmcat and tcpinc working groups
  - Author of multiple RFCs, e.g. LEDBAT (RFC6817) and ConEx (RFC7786, RFC7837)
- Project coordinator of the EU H2020 **Measurement and Architecture for a Middlebox'ed Internet (MAMI) project**
  - See mami-project.eu
  - Twitter: @mamiproject

Overview
QUIC

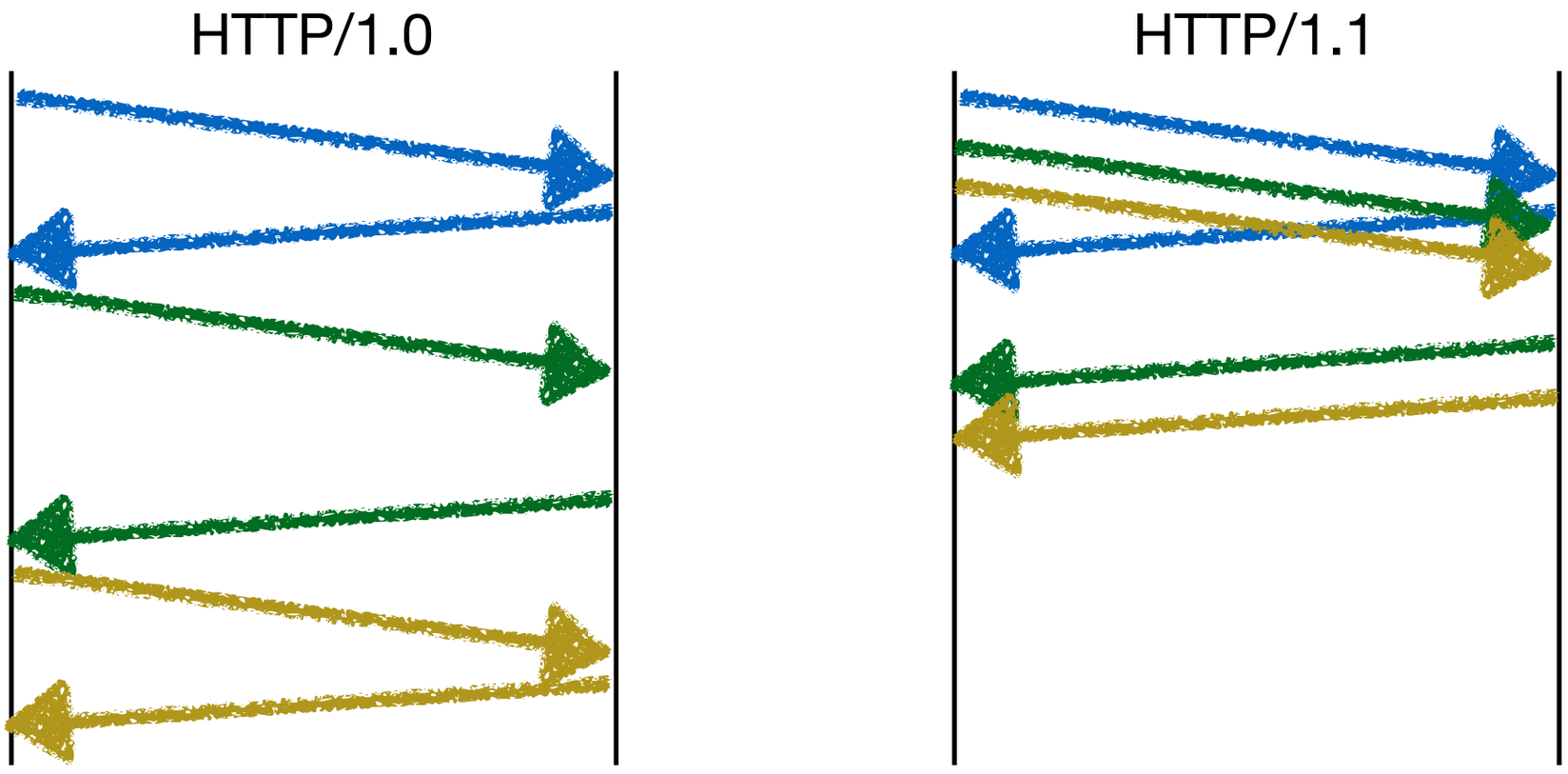
- New transport protocol, e.g. an alternative to TCP
- Currently under standardization in the IETF (since Nov’16)
- Optimized for HTTP and latency reductions
- Originally developed by Google together with SPDY
- Designed to be implemented in user space, e.g. in your browser

Overview HTTP/2

- Published as IETF RFC7540 in May 2015
- HTTP/2 is based on SPDY as originally proposed by Google
- About 250K domains support HTTP/2[1)]
  - Deployment driven by big hosting providers
- New features to reduce latency of page load time
  - Header compression (HPACK)
  - Server Push
  - Multiplexing of multiple requests over 1 TCP connection

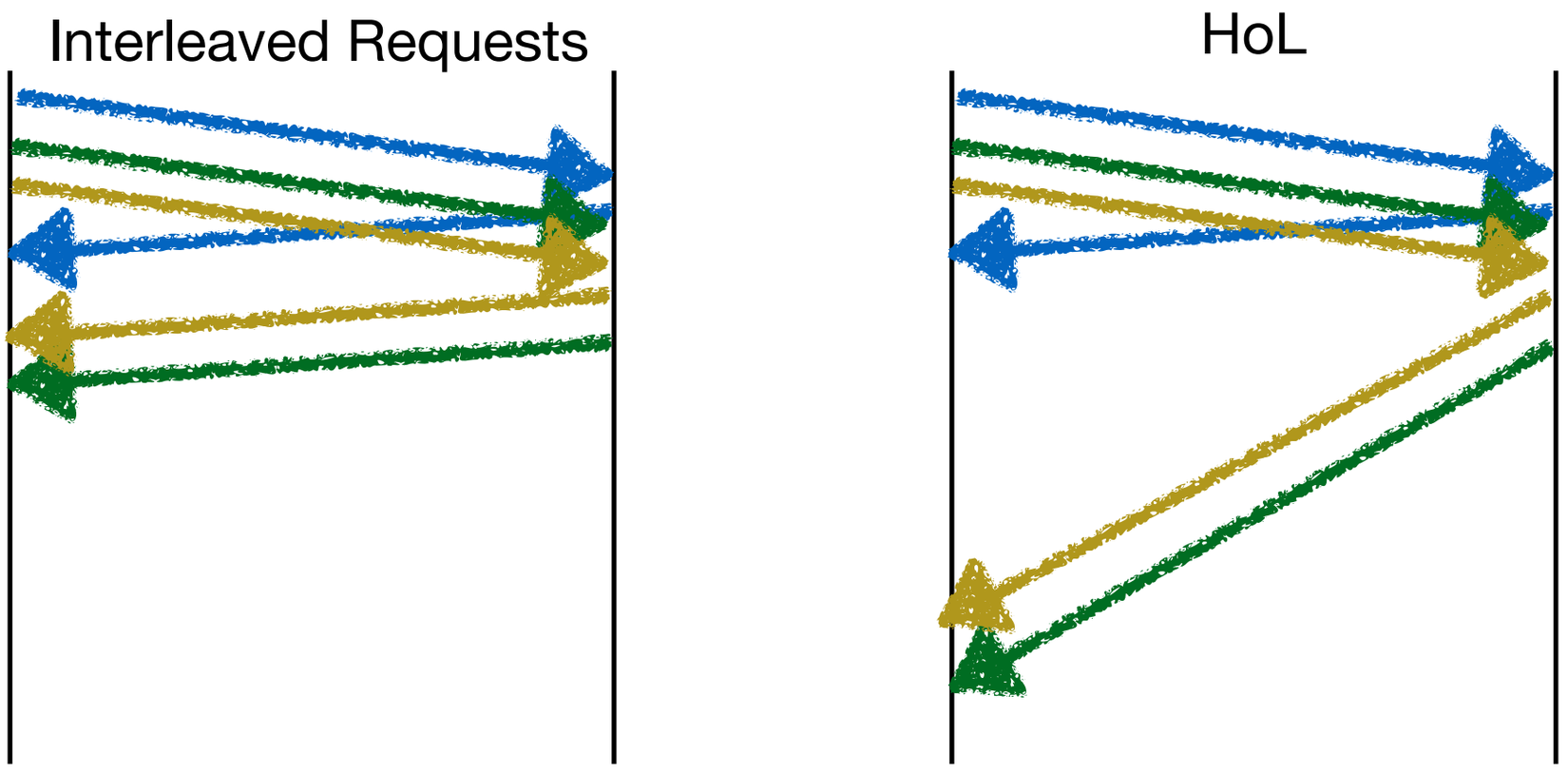
1) e.g. see isthewebhttp2yet.com

HTTP/1.1 — Pipelining



- With pipelining multiple requests can be sent at once
  - ➡ All requests and responses can be sent over the same TCP connection

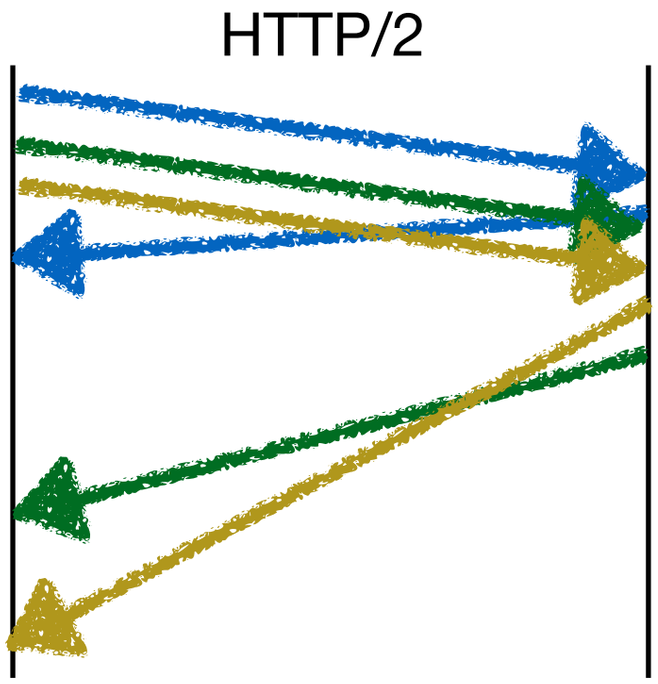
HTTP — Head of Line Blocking (HoL)



- Large resources can delay later, more important resources

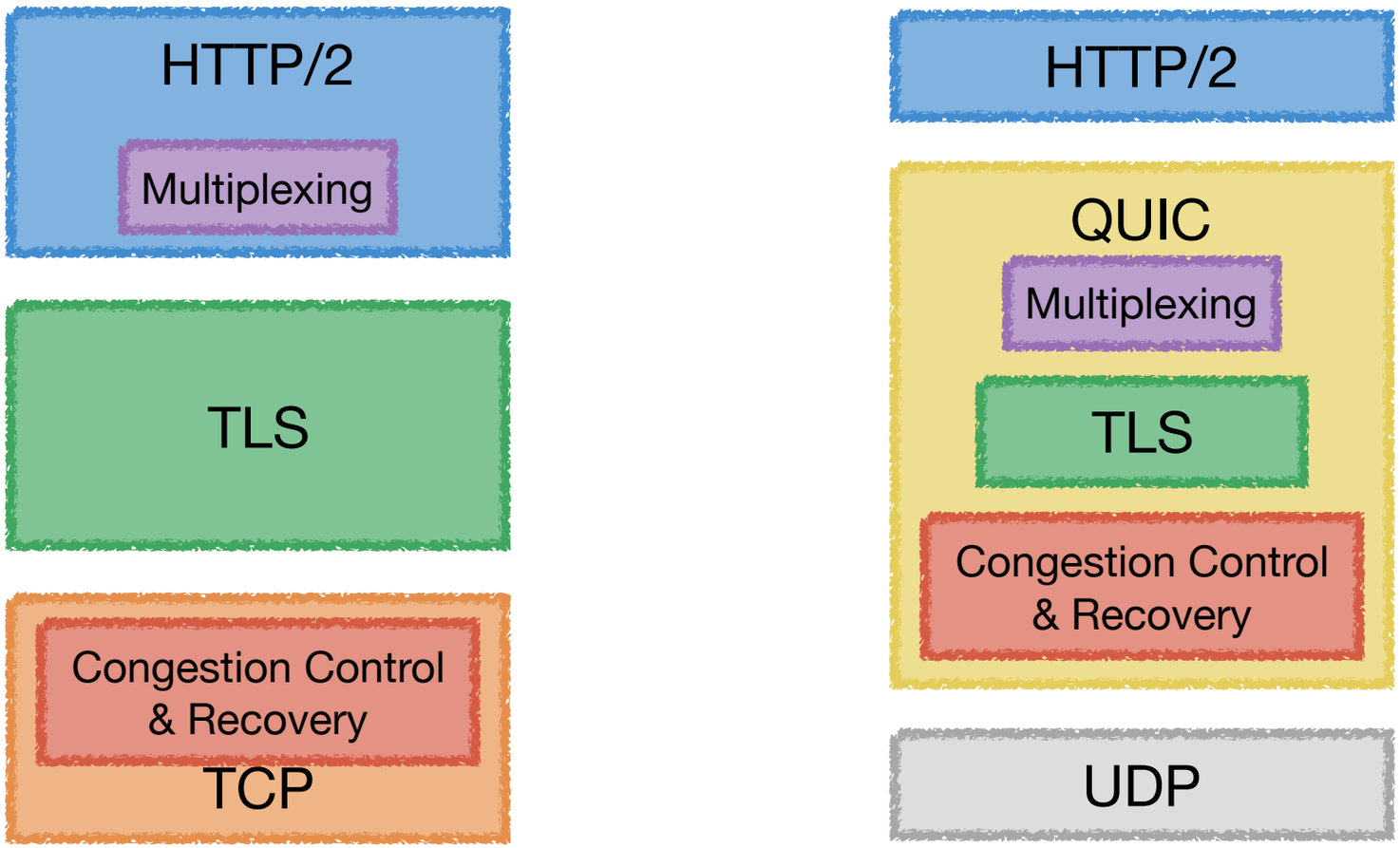
  ➡ Head-of-Line (HoL) blocking is still possible

HTTP/2 — Stream Multiplexing



- Requests are send on independent streams in frames
  - ➡ Loss or reordering can lead to Head-of-Line (HoL) blocking in the transport layer/TCP

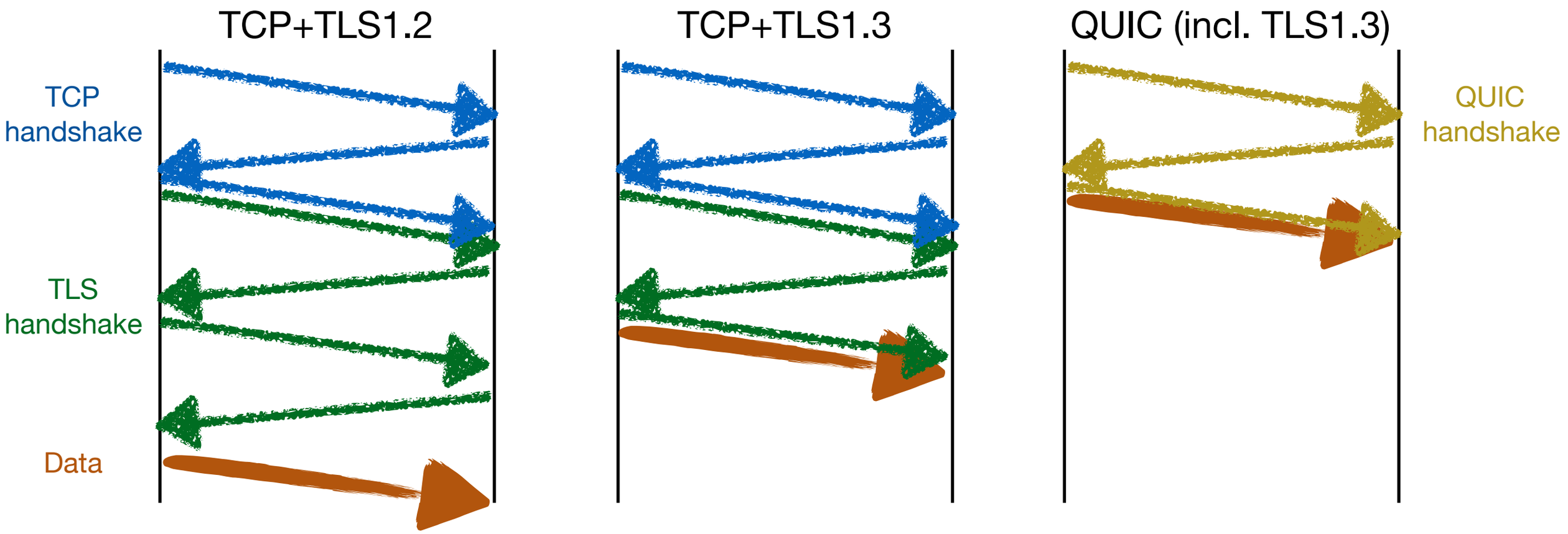
HTTP/2 over QUIC



- Stream multiplexing avoids HoL of independent data resources
- Always authenticated and payload is fully encrypted

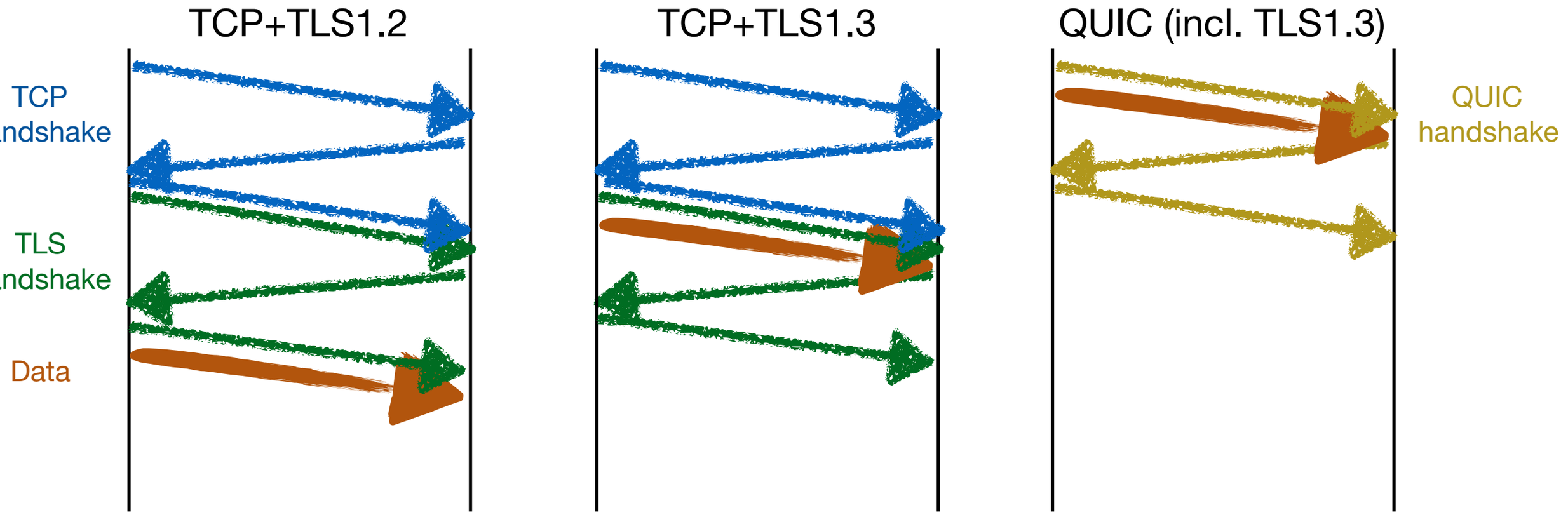
➡ **Stream & other control information (for e.g. recovery) is not visible to the network anymore!**

QUIC —
Low Latency Session Establishment



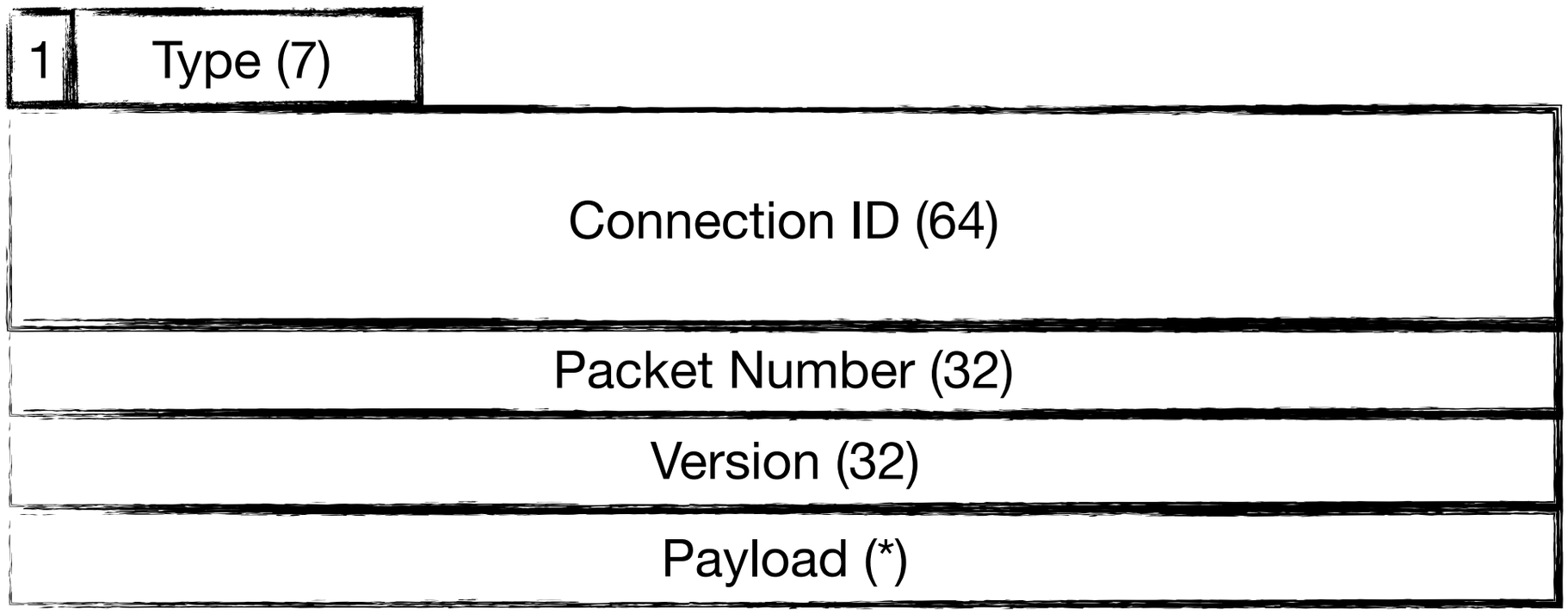
- QUIC combines the transport and crypto handshake
- Handshake packets are not encrypted but verified later

QUIC — 0-RTT Session Resumption



- QUIC can send encrypted payload data in the first RTT on session resumption to a previously connected server
- Care must be taken as QUIC/TLS1.3 0-RTT data are not protected against reply attacks

The QUIC wire image — Long header format (handshake & 0-RTT)



- Connection ID for migration and resilience to NAT rebinding
- Packet number is initiated to a random values and is unique within a connection
  - Packet Number also provides input to decryption

QUIC Long Header Types

Type	Name
0x01	Version Negotiation
0x02	Client Initial
0x03	Server Stateless Retry
0x04	Server Cleartext
0x05	Client Cleartext
0x06	0-RTT **Protected**
0x07	1-RTT **Protected** (key phase 0)
0x08	1-RTT **Protected** (key phase 1)

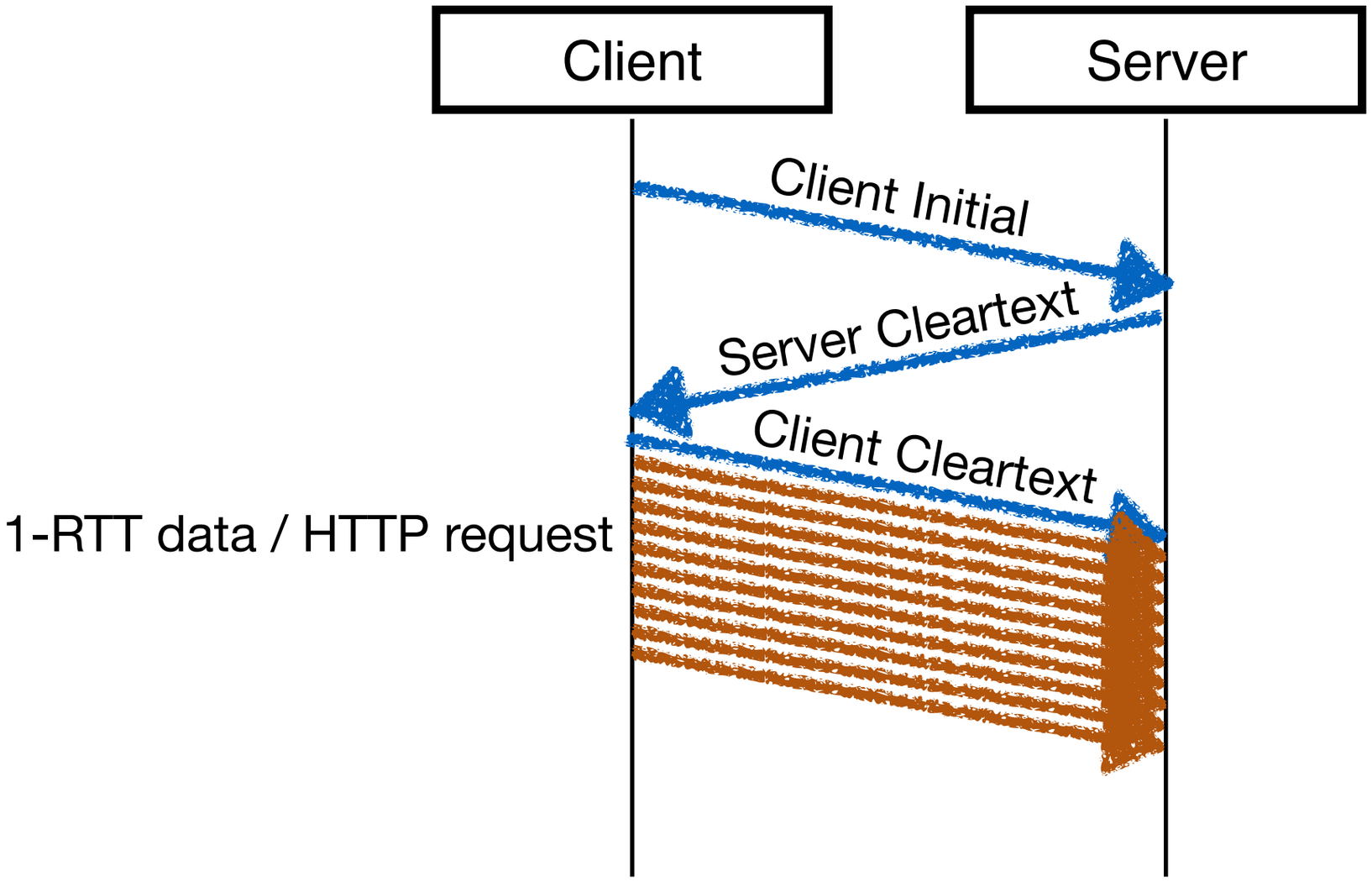
- Cleartext packets for Client Initial, Server Stateless Retry, and Server/Client handshake
- Encrypted payload for 0-RTT and 1-RTT data

QUIC
Version Negotiation

- Basically everything can change between QUIC version!

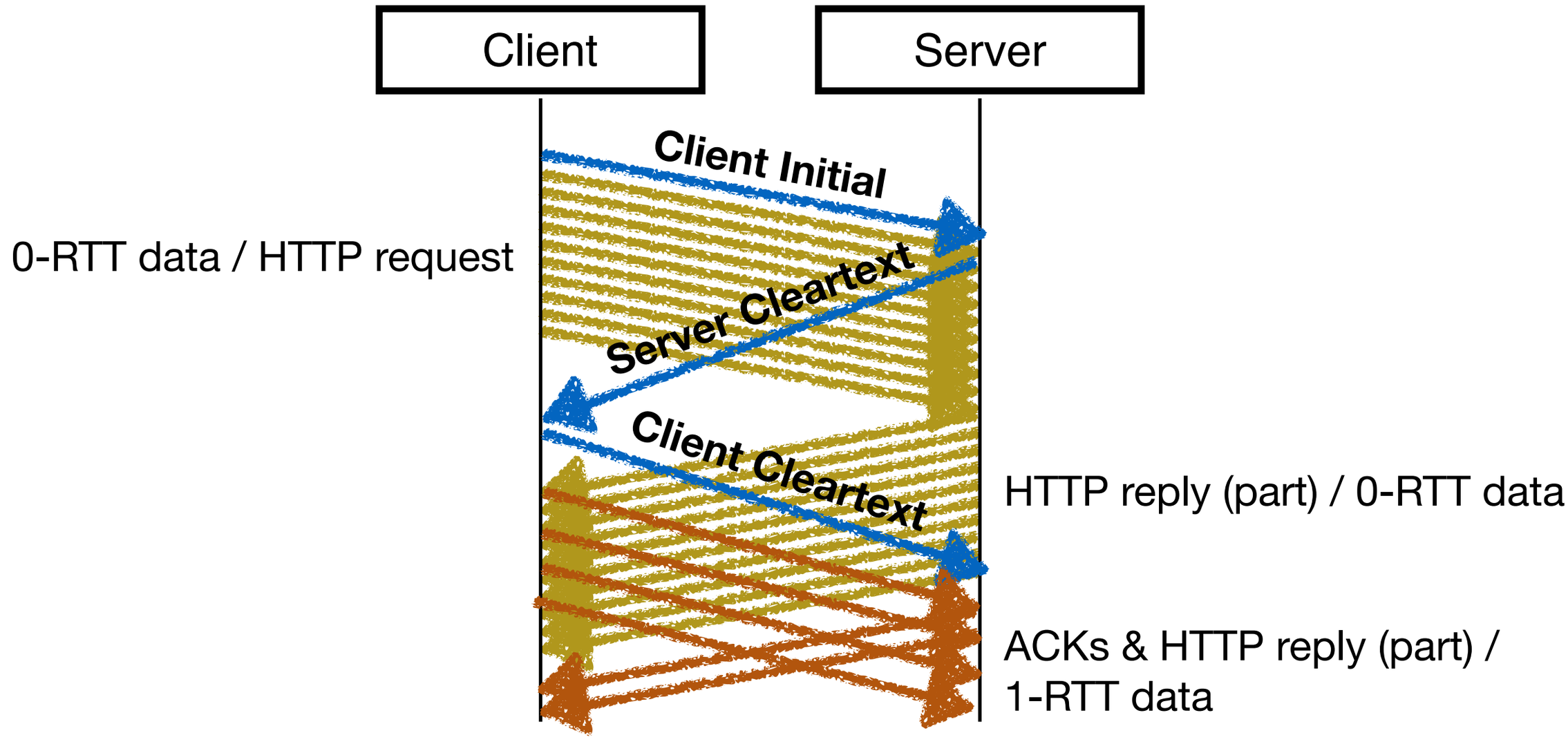
- Only to make version negotiation work, between different versions the following things need to remain the same:
  - the location of the header form flag,
  - the location of the Connection ID flag in short headers,
  - the location and size of the Connection ID field in both header forms,
  - the location and size of the Version field in long headers, and
  - the location and size of the Packet Number field in long headers.

QUIC Handshake



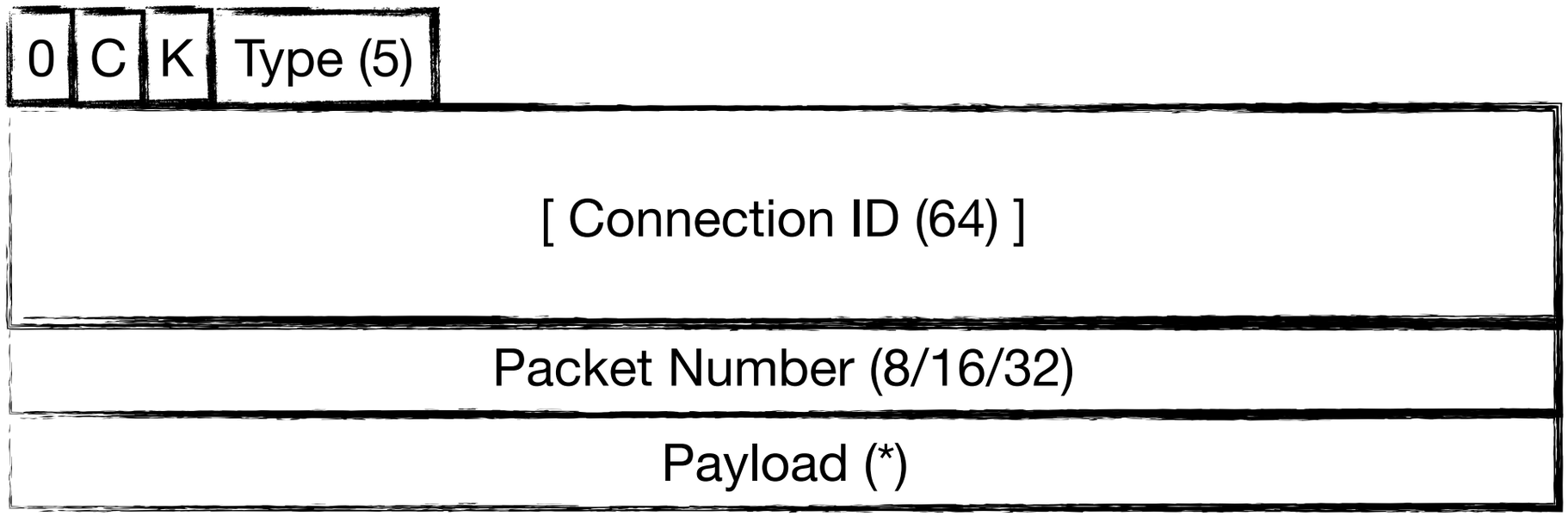
- Copy of the Client Initial and Server Cleartext are included in the encrypted part to verify content

QUIC Handshake — 0-RTT Session Resumption



- Client can send an initial window of data (10 packets) together with Client Initial

The QUIC wire image — Short header format (only 1-RTT)



- Connection ID is optional as indicated by the C flag
- Packet Number length is indicated by the header type
- K bit indications key phase for decryption
- Payload is always encrypted

QUIC wire image — Measurements and Monitoring

- Packet Number can be utilized to measure **packet loss (or reordering) so far** (from the sender to the observation point)
  - ➡ Packet number is monotonously increasing
  - ➡ But gap can also be introduced by the sender
  - ➡ Retransmissions and ECN congestion indications are not visible to the path (to estimate whole-path congestion)

- **Round-Trip Time (RTT)** can be estimated during the handshake
  - ➡ No easy was to correlate two packets in both direction during the rest of the connection

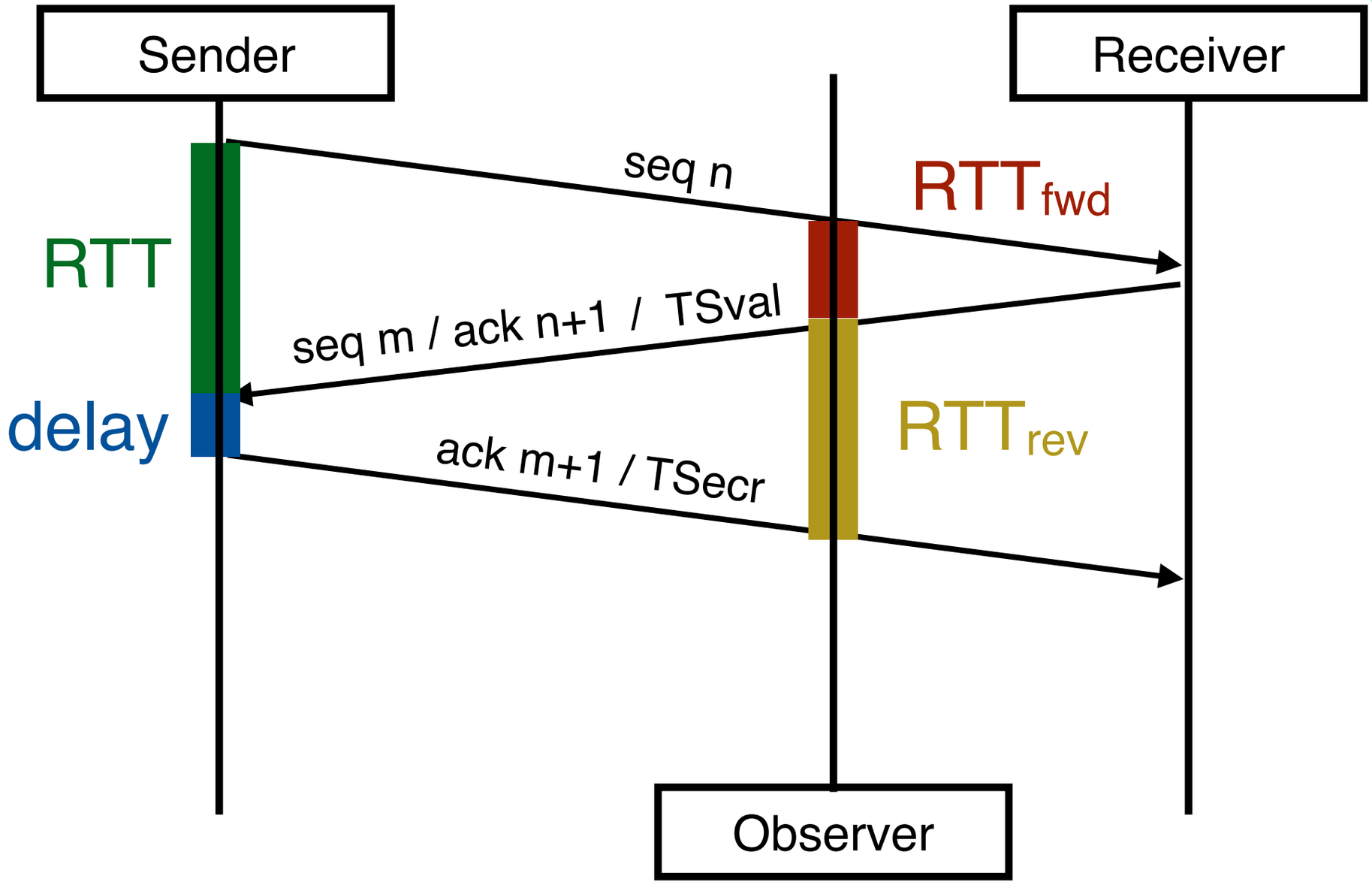
HTTP/2 and QUIC

- Both protocols are optimized for ***latency reductions*** by
  - multiplexing to avoid Head-of-line blocking,
  - overhead reduction (HPACK), and
  - optimizations in the in the handshake (QUIC and TLS).

- QUIC is currently ***still under development*** in the IETF
  - QUIC is a general purpose transport protocol optimized for HTTP/2
  - QUIC is always encrypted to enhance user privacy and avoid packet mangling
  - QUIC wire image is still under discussion to support basic monitoring support

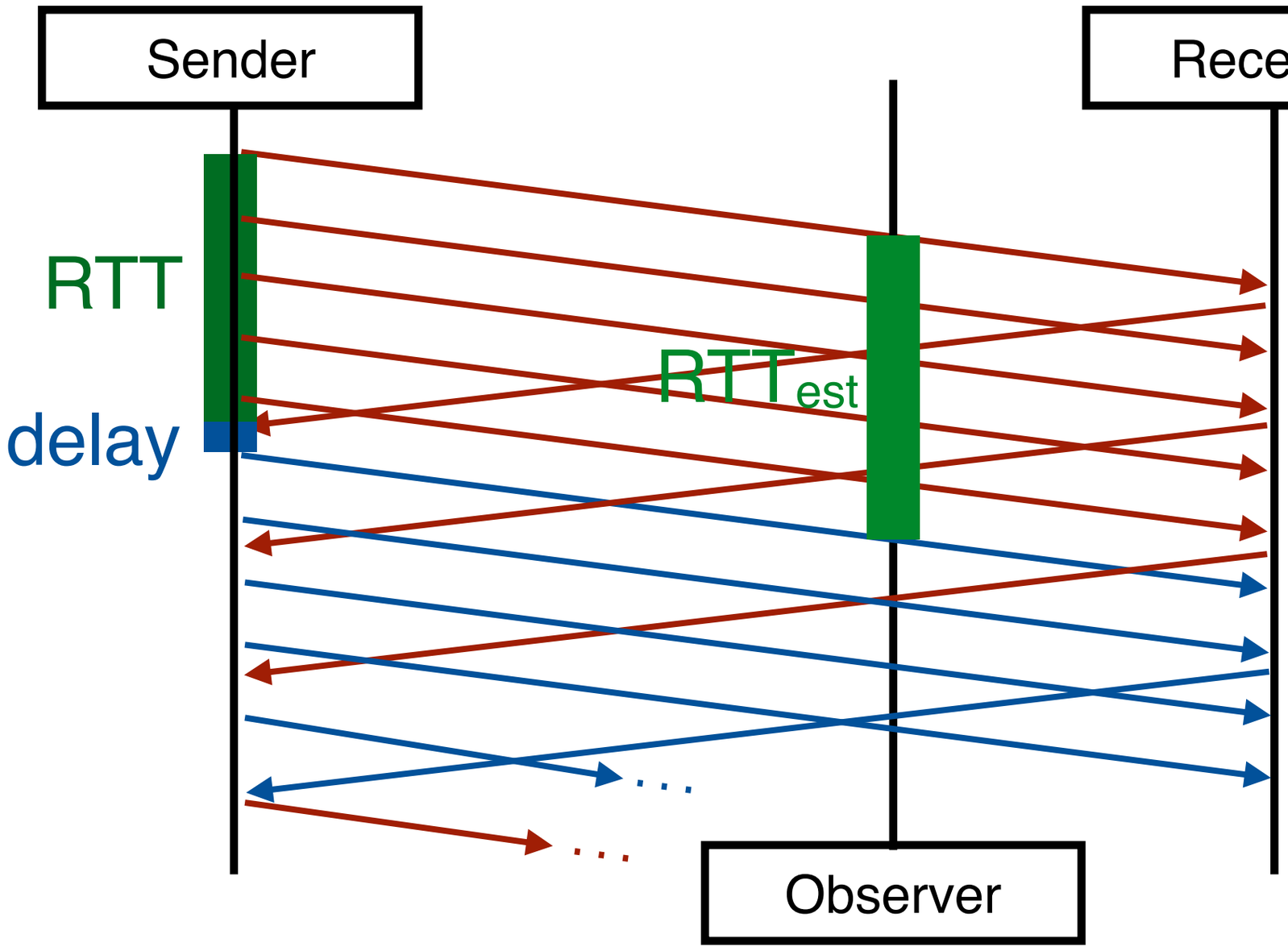
Backup

RTT estimation with TCP



- Use of SEQ# and ACK# and/or TCP Timestamp Option

Proposed Latency Spin Bit for QUIC



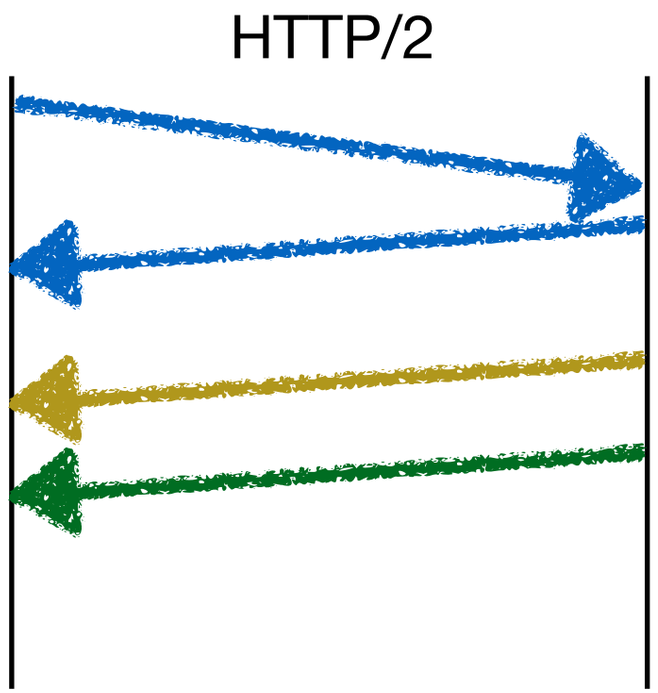
$t_{0\rightarrow1} - t_{1\rightarrow0}$

$t_{1\rightarrow0} - t_{0\rightarrow1}$

- Spin bit changes once per RTT

  ➡ enables observer in the network to take one RTT sample per RTT

HTTP/2 — Server Push



- Server can push resources with request