Quic Extension for Reporting Ack Receive Timestamps

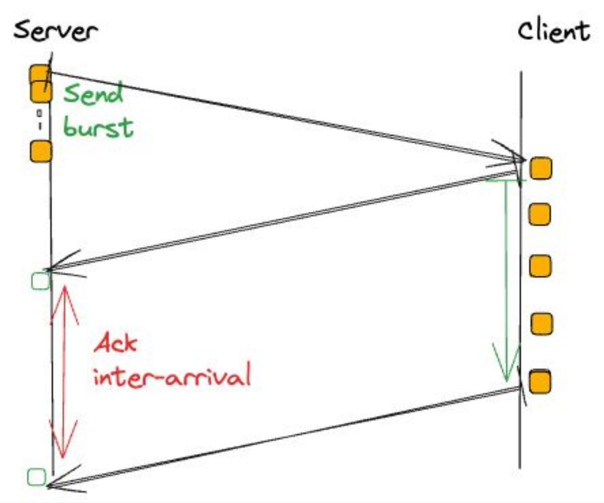
Meta Inc.

Motivation

QUIC congestion control [RFC9002] supports sampling round-trip time (RTT) by measuring the time from when a packet was sent to when it is acknowledged.

- meas. granularity depends on ack frequency (~20Pkts)

- hides n/w fluctuations at short time-scales (due to cell/wifi scheduling/congestion)

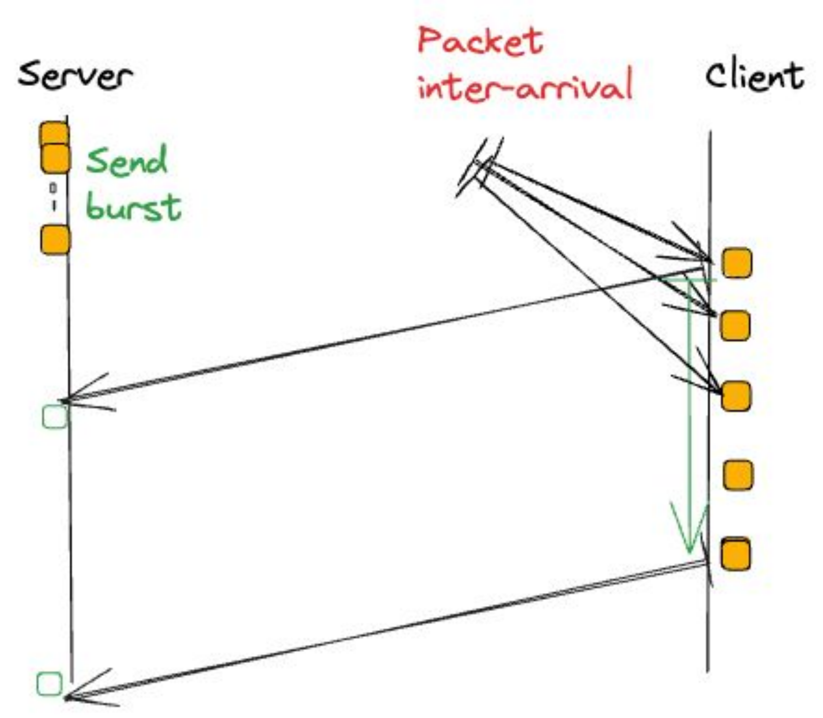


Motivation

Precise delay signals measured via packet receive timestamps have the potential to

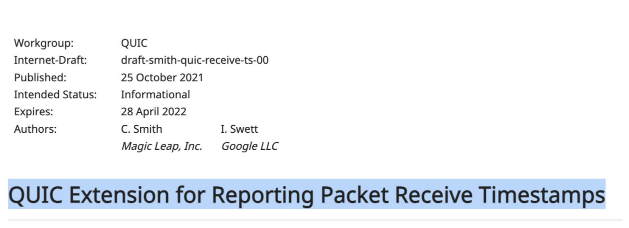
- improve accuracy of ***network bandwidth measurements and effectiveness of congestion control***, especially for latency-critical applications such as real-time video conferencing or game streaming
    - E.g. as used in WebRTC congestion control algorithm described in [I-D.ietf-rmcat-gcc]

- Capture short time-scale fluctuations in network conditions, to ***detect wireless last-mile network congestion***.



Draft Proposal Outline: Extension Negotiation

https://www.ietf.org/archive/id/draft-smith-quic-receive-ts-00.html



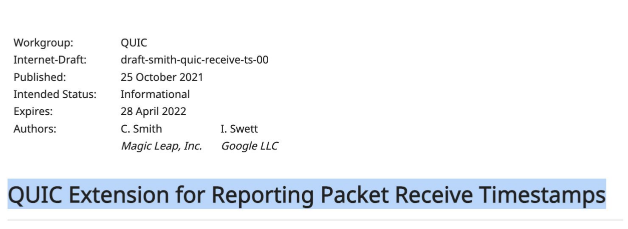
Workgroup:	QUIC	
Internet-Draft:	draft-smith-quic-receive-ts-00	
Published:	25 October 2021	
Intended Status:	Informational	
Expires:	28 April 2022	
Authors:	C. Smith *Magic Leap, Inc.*	I. Swett *Google LLC*

QUIC Extension for Reporting Packet Receive Timestamps

max_receive_timestamps_per_ack	A variable-length integer indicating that the sending endpoint would like to receive **ACK_RECEIVE_TIMESTAMPS** frames from the peer containing no more than the given maximum number of receive timestamps.
receive_timestamps_exponent	A variable-length integer indicating the exponent to be used when encoding and decoding timestamp delta fields in **ACK_RECEIVE_TIMESTAMPS** frames sent by the peer (see Section 5.1). If this value is absent, a default value of 0 is assumed (indicating microsecond precision). Values above 20 are invalid

ACK_RECEIVE_TIMESTAMPS Frame

```
ACK_RECEIVE_TIMESTAMPS Frame {
  Type (i) = TBD
  // Fields of the existing ACK (type=0x02) frame:
  Largest Acknowledged (i),
  ACK Delay (i),
  ACK Range Count (i),
  First ACK Range (i),
  ACK Range (..) ...,
  // Additional fields for ACK_RECEIVE_TIMESTAMPS:
  Timestamp Range Count (i),
  Timestamp Ranges (..) ...,
 }
```



Workgroup:	QUIC	
Internet-Draft:	draft-smith-quic-receive-ts-00	
Published:	25 October 2021	
Intended Status:	Informational	
Expires:	28 April 2022	
Authors:	C. Smith *Magic Leap, Inc.*	I. Swett *Google LLC*

QUIC Extension for Reporting Packet Receive Timestamps

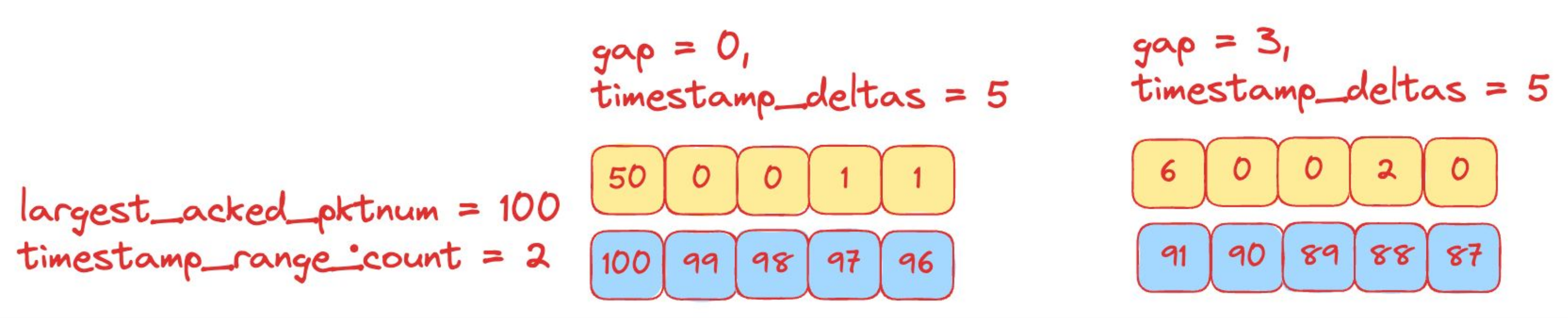
Endpoints which send ACK_RECEIVE_TIMESTAMPS frames must determine a value, ***receive_timestamp_basis***, relative to which all receive timestamps for the session will be reported

The value of *receive_timestamp_basis* MUST be less than the smallest receive timestamp reported, and MUST remain constant for the entire duration of the session

Timestamp Ranges Encoding

Each Timestamp Range describes a series of contiguous packet receive timestamps in descending sequential packet number (and descending timestamp) order.

Gap	A variable-length integer indicating the largest packet number in the Timestamp Range (as the difference between the largest acked packet, or the smallest packet of the previous range)
Timestamp Delta Count	A variable-length integer indicating the number of Timestamp Deltas in the current Timestamp Range.
Timestamp Deltas	Variable-length integers encoding the receive timestamp for contiguous packets in the Timestamp Range in descending packet number



ACK_RECEIVE_TIMESTAMPS Frame: Meta’s implementation

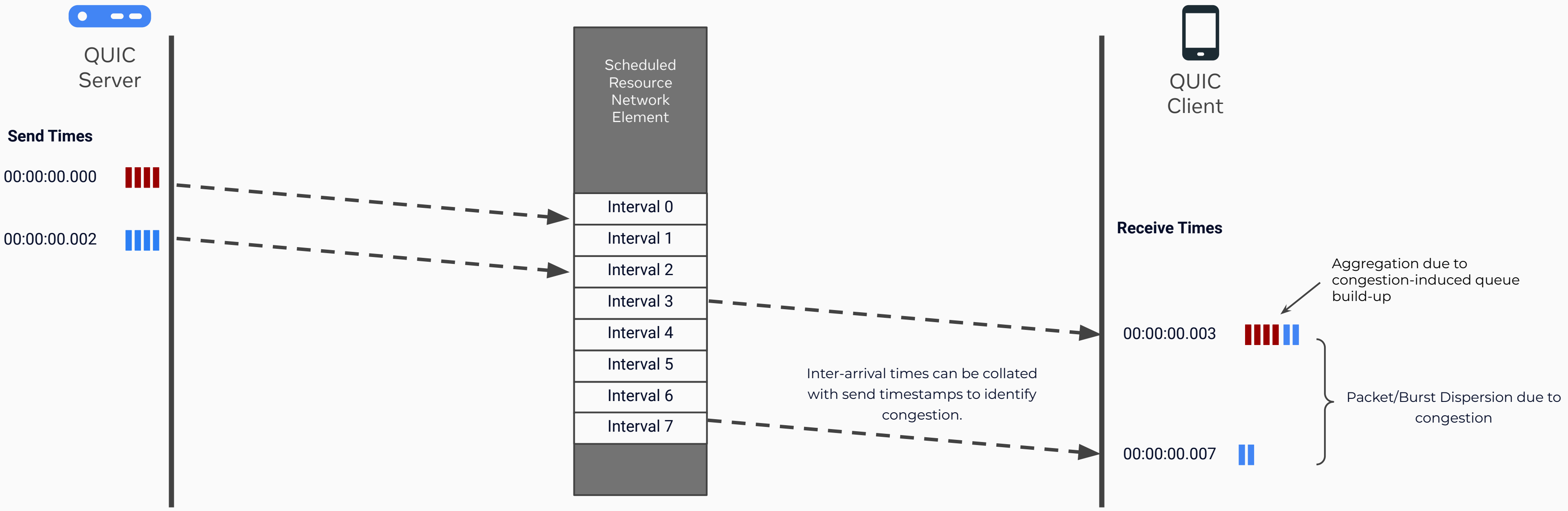
```
ACK_RECEIVE_TIMESTAMPS Frame {
  Type (i) = TBD
  // Fields of the existing ACK
(type=0x02) frame:
  Largest Acknowledged (i),
  ACK Delay (i),
  ACK Range Count (i),
  First ACK Range (i),
  ACK Range (..) ...,
  // Additional fields for
ACK_RECEIVE_TIMESTAMPS:
  Latest Received Packet Num (i)
  Latest Received Packet Time Delta (i)
  Timestamp Range Count (i),
  Timestamp Ranges (..) ...,
 }
```

Rx timestamps are NOT generated for out-of-order packets.

Latest received pkt_num/time_delta received surfaces latest was an out-of-order packet.

Allows for more accurate RTT measurements

Wireless Last-mile Congestion Detection



QUIC Server

Send Times

00:00:00.000

00:00:00.002

Scheduled Resource Network Element

Interval 0

Interval 1

Interval 2

Interval 3

Interval 4

Interval 5

Interval 6

Interval 7

Inter-arrival times can be collated with send timestamps to identify congestion.

QUIC Client

Receive Times

Aggregation due to congestion-induced queue build-up

00:00:00.003

Packet/Burst Dispersion due to congestion

00:00:00.007

Queues build up for non-serviced clients when waiting for resources.
This queuing results in packet aggregation and dispersion.

Use cases @ Meta

Congestion Detection:

- Explicit signal for congestion can improve QUIC congestion control algorithms.
- Adaptive bitrate (ABR) optimizations to better adapt to congestion.

Bandwidth Estimation:

- Fine-grained, short time-scale BW estimates help latency-sensitive applications

Discussion and Next-steps

Proposal: extend current draft to support encoding out-of-order packet ranges (and time-of-arrival deltas) in the ACK_RECEIVE_TIMESTAMPS frame

- useful for more accurate RTT measurements,
- understanding the magnitude and impact of potentially high out-of-order packet scenarios, e.g Quic Direct Server Return (DSR).