# QUIC Recovery Max Ack Delay

QUIC Interim 2018-01, Melbourne

## Updates since 07

Many editorial issues fixed

Pacing is now recommended without a specific approach

Max Ack Delay has been introduced

## Filtering RTT measurements

```
UpdateRtt(latest_rtt, ack_delay):
    // min_rtt ignores ack delay.
    min_rtt = min(min_rtt, latest_rtt)
    // Adjust for ack delay if it's plausible.
    if (latest_rtt - min_rtt > ack_delay):
      latest_rtt -= ack_delay
      // Only save into max ack delay if it's used
      // for rtt calculation and is not ack only.
      if (!sent_packets[ack.largest_acked].ack_only)
        max_ack_delay = max(max_ack_delay, ack_delay)
```

## Using Max Ack Delay

Tail Loss Probe and RTO now include max ack delay

<u>TLP:</u>
```
max(1.5 * smoothed_rtt + max_ack_delay,
    kMinTLPTimeout)
```

<u>RTO:</u>
```
max(smoothed_rtt + 4 * rttvar +
    max_ack_delay, kMinRTOTimeout)
```

QUIC

## Explicitly Communicated Max Ack Delay

Pros:

- May be able to avoid spurious timeouts

Cons:

- Adds a transport param that cannot be encrypted from client to server.

Issue [#981]

## Eliminating MinRTO

TCP Max Ack Delay proposal includes eliminating MinRTO

Pros:

- Decreases tail latency

Cons:

- Increases chance of spurious RTO

Issue #1017

## Plan: Gather Data

Google intends to conduct these experiments, and I would encourage others to as well:

1. TCP style (Ignore ack delay entirely)
2. Explicitly communicate max ack delay
3. Remove MinTLP - Currently 10ms
4. Remove MinRTO - Currently 200ms

Hope to present transport and application metrics in London