Google

TCP Low Latency Option

draft-wang-tcpm-low-latency-opt-00

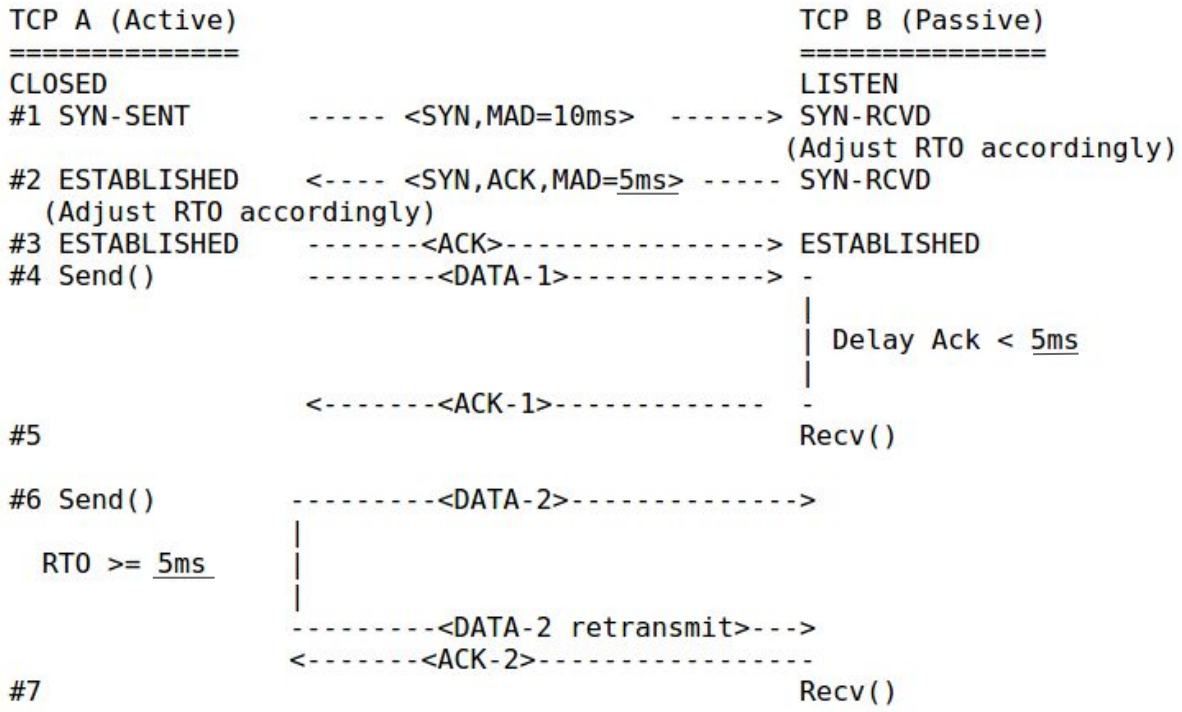
Wei Wang
Neal Cardwell
Yuchung Cheng
Eric Dumazet

IETF 99: Prague, July 17, 2017

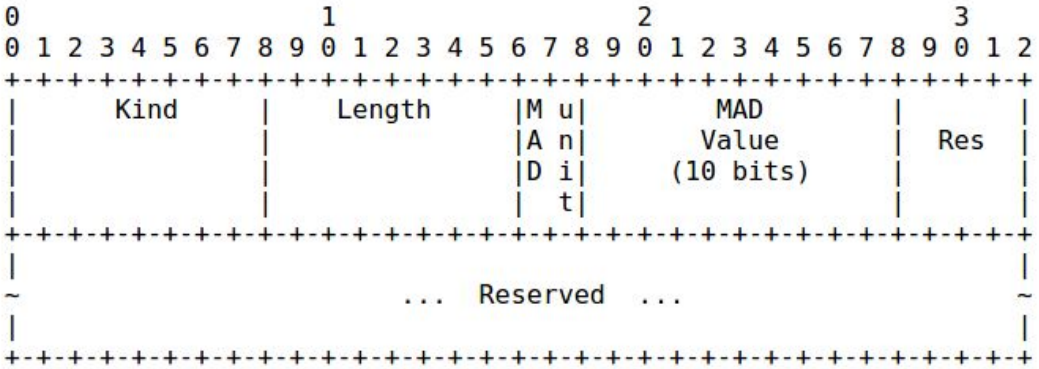
Motivation: lower latency, higher throughput

- Datacenters with commodity 10Gbps Ethernet: RTT <100 us
- Outdated fixed parameters:
  - RFC1122: Delayed ACKs: typical delays: 40 ms .. 200 ms [400x RTT]
  - RFC6298: minimum RTO of 1 sec [10,000x RTT]
  - RFC7323: TCP Timestamps option has granularity of 1 ms .. 1 sec [10x RTT]
- Solution:
  - Advertise hints of related parameters used on the local side during connection establishment
  - Pick up the hint and do corresponding adjustment on the remote side

3-way handshake flow chart

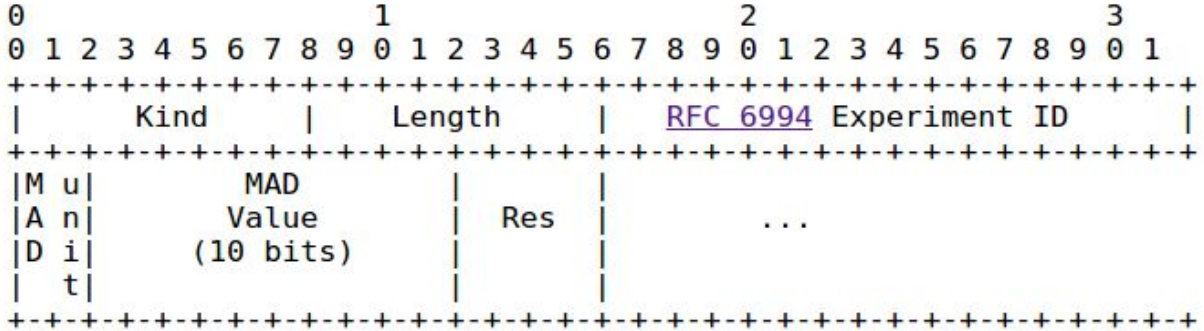


TCP Low Latency Option




```
Kind:       1 byte: value = IANA-assigned option number
Length:     1 byte: value = 4 (or longer in later versions)
MAD unit:   2 bits: indicates time unit for MAD value:
                    0: reserved
                    1: milliseconds
                    2: microseconds
                    3: nanoseconds
MAD value:  10 bits: indicates MAD value set on the host:
                     1 ... 1023: MAD value in the given units
                     0: no MAD value is specified
Reserved:   N>=4 bits: value = 0
```

TCP Low Latency Option with Experimental ID



Kind: 1 byte: value = 254
Length: 1 byte: value = 6 (or longer in later versions)
Experiment ID: 2 bytes: value = 0xF990
MAD unit: 2 bits: indicates time unit for MAD value:
0: reserved
1: milliseconds
2: microseconds
3: nanoseconds
MAD value: 10 bits: indicates MAD value set on the host:
1 ... 1023: MAD value in the given units
0: no MAD value is specified
Reserved: N>=4 bits: value = 0

Configuring Maximum Ack Delay (MAD)

- An implementation that supports the maximum ACK delay parameter MUST provide a user API to configure it for a specific connection or all TCP connections.
  - In Linux, we are proposing 2 APIs to configure MAD:
    - Ip route command
    - setsockopt()
- the implementation SHOULD use a value as close as possible to the user-specified value as the maximum timeout for the delayed ACK of the specified TCP connections.
- Note that the actual maximum delayed ACK timeout of the connection may be larger than the actual user specified value because of implementation constraints (e.g. timer granularity limitations).

Announcing Maximum Ack Delay (MAD)

- The maximum ACK delay is announced to the remote TCP endpoint by including a Low Latency option with a non-zero MAD value in the SYN or SYN/ACK packet.
  - Normally, both active and passive side should advertise their own MAD value.
  - If active side does not announce its MAD value, passive side will not announce its own MAD value.
- If specified, the MAD value in the Low Latency option MUST be set to the implementation's actual delayed ACK timeout for the connection.

Adjusting TCP retransmission timeouts

- The data sender MAY use the MAD value advertised by the receiver to adjust the sender's RTO calculation.
  - RTO <- max(SRTT + max(G, K*RTTVAR), 1 second) /* [RFC6298] */

    RTO <- SRTT + max(G, K*RTTVAR) + max(G, max_ACK_delay)

  - In [draft-ietf-tcpm-rack] when computing PTO:

    If an SRTT estimate is available:

    PTO = 2 * SRTT

    Else:

    PTO = initial RTO of 1 sec

    If FlightSize = 1:

    PTO = max(PTO, 1.5 * SRTT + WCDelAckT) max_ACK_delay

    PTO = max(10ms, PTO)

    PTO = max(RTO, PTO)

Status

- Initial draft submitted to IETF: [draft-wang-tcpm-low-latency-opt-00]
- Second draft will include microsecond timestamp
- Maximum ACK Delay has been used in Google since Jul 2005
- Microsecond timestamp has been used in Google since Feb 2015
- Upstream Linux implementation under development
- Support for other platforms/OS ?

Thank you