BBR Drain Pacing Gain: a Derivation

September 2021
The Google BBR team

This document presents an analytic derivation of the BBR Drain state pacing gain constant.

Analytic derivation

In the Drain state, BBR aims to quickly drain any queue created in Startup by switching to a pacing_gain well below 1.0. Specifically, it uses a pacing_gain that is selected to try to drain the queue in one packet-timed round trip.

The following derivation shows that that pacing_gain that meets that goal is the pacing_gain that is the reciprocal of the cwnd_gain value used during Startup.

The derivation runs as follows.

Our goal is to have a BDP of data in flight in the network at the end of Drain:

```
inflight_at_end_of_drain = bdp = bw * min_rtt
```

We can model the amount of data in flight at the end of drain as the amount of data we start with, plus the data that we send during Drain, minus the data that is delivered during Drain:

```
inflight_at_end_of_drain =  starting_inflight + packets_sent_in_drain -
packets_delivered_in_drain
```

We can model the constituent terms as:

```
starting_inflight = BBRStartupCwndGain * bw * min_rtt

packets_sent_in_drain = BBRDrainPacingGain * bw * drain_time

packets_delivered_in_drain = bw * drain_time
```

We can also model the expected time spent in Drain, given that the goal is one packet-timed round trip:

```
drain_time = packet_timed_round_trip
           = min_rtt + queue_delay
           = min_rtt + queue_size/bw
           = min_rtt + (starting_inflight - bdp)/bw
           = min_rtt + (BBRStartupCwndGain * bw * min_rtt - bw*min_rtt)/bw
           = min_rtt + (BBRStartupCwndGain - 1 ) * bw * min_rtt /bw
           = min_rtt + (BBRStartupCwndGain - 1 ) * min_rtt
           = min_rtt + (BBRStartupCwndGain - 1 ) * min_rtt
```

Combining the above equations we get:

```
inflight_at_end_of_drain =
  starting_inflight + packets_sent_in_drain - packets_delivered_in_drain =
  BBRStartupCwndGain * bw * min_rtt +
  BBRDrainPacingGain * bw * drain_time  -
  bw * drain_time
  =
  BBRStartupCwndGain * bw * min_rtt +
  (BBRDrainPacingGain - 1) * bw * drain_time
```

We desire:

```
inflight_at_end_of_drain = bdp
```

Thus we can set:

```
bdp = inflight_at_end_of_drain =
  BBRStartupCwndGain * bw * min_rtt +
  (BBRDrainPacingGain - 1) * bw * drain_time
```

Solving for `BBRDrainPacingGain` we get:

```
bdp =
   BBRStartupCwndGain * bw * min_rtt +
   (BBRDrainPacingGain - 1) * bw * (min_rtt + (BBRStartupCwndGain - 1 ) * min_rtt) =
   BBRStartupCwndGain * bw * min_rtt +
   (BBRDrainPacingGain - 1) * bw * (min_rtt + (BBRStartupCwndGain - 1 ) * min_rtt) =
```

```
   BBRStartupCwndGain * bw * min_rtt +
   (BBRDrainPacingGain - 1) * bw * (min_rtt + BBRStartupCwndGain * min_rtt - min_rtt) =
   BBRStartupCwndGain * bw * min_rtt +
   (BBRDrainPacingGain - 1) * bw * (BBRStartupCwndGain * min_rtt) =
   BBRStartupCwndGain * bw * min_rtt +
   (BBRDrainPacingGain - 1) * bw * (BBRStartupCwndGain * min_rtt)

bdp = BBRStartupCwndGain * bdp + (BBRDrainPacingGain - 1) * BBRStartupCwndGain * bdp

1 = BBRStartupCwndGain + (BBRDrainPacingGain - 1) * BBRStartupCwndGain

1 = BBRStartupCwndGain + (BBRDrainPacingGain - 1) * BBRStartupCwndGain

1 / BBRStartupCwndGain = 1 + BBRDrainPacingGain - 1

1 / BBRStartupCwndGain = BBRDrainPacingGain

BBRDrainPacingGain = 1 / BBRStartupCwndGain
```

That shows why we set the Drain mode pacing_gain to be the reciprocal of the Startup cwnd_gain, in order to attempt to drain the queue in one packet-timed round trip.