BBR Congestion Control Draft

draft-ietf-ccwg-bbr-03

Internet Draft Editors:

Neal Cardwell (Google), Ian Swett (Google), Joseph Beshay (Meta)

Speaker: Ian Swett

Outline

Overview

- Outline recent BBR Internet Draft updates
- Summarize open pull requests and open issues

Goals for this talk:

- Provide a road map for...
  - Readers of the draft
  - Implementers of BBR reading the draft
  - Members of the CCWG/ICCRG community who would like to contribute
- Inviting the community to…
  - Read the draft
  - Contribute to the draft

Overview of draft-ietf-ccwg-bbr

- BBR was adopted as a CCWG WG item in October 2024
- Intended status: experimental RFC
- IETF CCWG members are collaborating on github:
  - https://github.com/ietf-wg-ccwg/draft-ietf-ccwg-bbr
- Latest published revision is at:
  - https://datatracker.ietf.org/doc/draft-ietf-ccwg-bbr/
- Latest editor's copy (with hot-off-the-press changes not in published revisions above):
  - https://ietf-wg-ccwg.github.io/draft-ietf-ccwg-bbr/draft-ietf-ccwg-bbr.html
- Draft editors:
  - Neal Cardwell (Google)
  - Ian Swett (Google)
  - Joseph Beshay (Meta)

Changes in draft-ietf-ccwg-bbr-03: summary

- Previous version was: draft-ietf-ccwg-bbr-02, discussed at IETF 122 in Mar, 2025
- Diffs: [text diff between draft-ietf-ccwg-bbr-02 and draft-ietf-ccwg-bbr-03]
- Main changes between draft-ietf-ccwg-bbr-02 and draft-ietf-ccwg-bbr-03:
  - Documented stance on ECN (here; issue #7): specific CE response not specified, but if connection's packets are sent with ECT(0) or ECT(1) then connection "MUST treat any CE marks as congestion."
  - Described why BBR draft is experimental (here; issue #51): no specific ECN spec; various rate sampling limitations
  - Moved delivery rate and RTT sampling specs to a new top-level Sec 4. Input Signals section to make them easier to find and read
  - Define "inflight" and use it more precisely and consistently #41 #45
  - … a number of other editorial improvements for clarity/readability

Pending Experiments: 2 Open PRs: minor changes

First 2 Open PRs; both are algorithm changes waiting for performance data from experiments:

- #6 Use consistent value for drain pacing gain which matches derivation doc
  - Proposes changing BBR DRAIN gain from 1/2.89 = 0.35 to 1/2 = 0.5
  - To match analytical derivation of DRAIN pacing gain, which derives 1/2 = 0.5
  - Have some old A/B experiment data from Linux TCP YouTube experiments:
    - Unclear if there are statistically significant performance regressions
    - Planning to re-run an experiment to ensure there are no regressions
  - We'd appreciate performance data comparing drain_gain vals from anyone who's able
- #5 Remove BBR.ack_phase from pseudocode
  - A minor algorithm simplification
  - Has one implementation (mvfst QUIC BBR2)
  - We'd like
    - A second implementation
    - Internet performance data to verify there is no performance regression

PR: Delivery rate sampling and restarting from idle

An open PR:

- Clarify Unacknowledged rather than inflight for rate sample #56
  - In Sec `4.1.2.2. Transmitting a data packet` of -03
  - **How to check for a connection "restarting from idle", in a transport/implementation-agnostic way?**
    - e.g.: connection is certain that no data packets are in the network
  - TCP-centric version was: `if (SND.NXT == SND.UNA)  /* no packets in flight yet? */`
  - (C.inflight == 0) is not sufficient, for subtle reasons
    - packets spuriously marked lost and later delivered would have incorrect timestamps that could cause significant bandwidth overestimation
  - Open for discussion; ideas are welcome

PR: Specifying application-limited check

An open PR:

- Fix pacing spec in "application-limited" section #47
  - In Sec `4.1.1.3. Tracking application-limited phases` of -03
  - **How to specify the application-limited check in transport/implementation-agnostic way?**
  - Current text in the draft is agreed to be not generic enough to cover all pacing implementations
  - Current proposal being discussed has evolved from suggestion by Christian Huitema:
    - "BBR detects that an application-limited phase has started for a connection when congestion control and pacing control would have allowed the connection to send data, and yet the connection is not currently sending data and has no data to send (i.e., no unsent/new data, no retransmit data, and no other data, such as encoded data blocks for forward error correction). The precise determination of this condition depends on how the connection uses mechanisms to implement pacing, batching, GSO/TSO/offload, etc."
  - Open for discussion; ideas/tweaks are welcome

Issue: generalization to non-TCP transports

Open issues:

- Bandwidth estimation still uses _seq names, which are TCP-centric
- Section 5.5.9 sounds very TCP-specific

The intent is to make the draft as transport agnostic as possible.
We are making progress but not done.

Goal: Ensure implementation of BBR across as many transports as possible

Non-Goal: Create universal approach for mapping any congestion control to any transport

Issue: ProbeRTT every 5 sec or 10 sec?

Open issue:

- Impact of increasing ProbeRTT interval to 10s?

Details:

- BBRv1 used a 10s ProbeRTT interval
  - This 10s value has many "air miles" for both TCP and QUIC
- BBRv2/BBRv3 currently use: "ProbeRTTInterval = 5 seconds"
  - Rationale: since BBRv2/BBRv3 ProbeRTT reduce cwnd to 0.5*estimated_BDP instead of 4 packets, they can achieve the same throughput as BBRv1 with 2x more frequent ProbeRTT
- Meta mvfst QUIC BBRv2 found:
  - Increasing ProbeRTT interval to 10s in internet facing connections improves application performance in initial experiments (vs 5s)
    - No increase in RTT estimates
    - No increase in retransmissions
    - Reduces p99 response time for requests by up to 15% for app traffic on busy links
- Gorry Fairhurst requested tests of impact on cross-traffic sharing bottlenecks with BBR
  - We are still planning A/B simulations of impact (BBR vs BBR; BBR vs CUBIC, etc)
  - Historical impact is that changing ProbeRTT 5s => 10s makes BBR **less** aggressive

Summary of other open Issues: part 1

Issues with technical implications:

- Application-limited detection depends on implementation of pacing #42
  - Related to the previously mentioned discussion on PR #47

- What is BBR.offload_budget for QUIC? #67
  - The description of the offloat_budget is specific to TCP. What's a good approach to expand it to accommodate QUIC?

Editorial questions:

- Definition of variables
  - Per-Packet and Connection State variables are spread across multiple Sections #68
  - notation for constants and member variables #63
  - C.delivered is undefined #60
  - What's the difference between the C and the BBR struct? #59
- Language clarification
  - What are filters? #65
  - What does tracking state per aggregate mean? #66

Summary of other open Issues: part 2

Editorial questions (cont.):

- Units: Bytes/Segments/Packets? Current proposal: Bytes
  - segment vs packet #62
  - Should delivered be called bytes_delivereds? #61
  - Should BBR inflight be described in bytes, packets or compatible with both? #27
- Pseudocode
  - clarify function parameters of BBRUpdateOnACK #58
  - BBR.extra_acked_filter is unused #64

Conclusion

- Inviting the community to...
  - Read the draft: draft-ietf-ccwg-bbr
  - Offer contributions/comments/edits, in whatever manner you prefer
- Thanks!

Details/Backups...

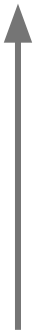
Changes in draft-ietf-ccwg-bbr: how to view

- To see recent changes, you can use several approaches, depending on your preference:
  - From the command line:
    - git clone https://github.com/ietf-wg-ccwg/draft-ietf-ccwg-bbr.git
    - cd draft-ietf-ccwg-bbr/
    - git log -p
  - From github:
    - Commits
    - Merged pull requests

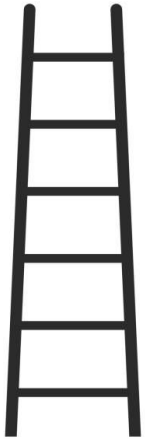
Thoughts about ways to contribute

- Contributions at any "rung of the ladders" below are welcome!
- The higher on the "ladders" (the more concrete/specific/tested the contribution is)...
  - The more useful to the BBR draft effort
  - Given editor time constraints, the more likely the eventual inclusion in the draft
- To finalize significant algorithm changes, we'd like to ultimately reach the top rung of the ladder
- Collaboration encouraged: e.g., idea from person A, implemented by person B, tested by sites B/C

Editorial changes:



- Github pull request with draft text
- Github issue describing the idea
- CCWG email/meeting suggestion



Technical algorithm changes:

- Multiple at-scale Internet deployments
- At-scale Internet deployment data
- Lab/simulation experiment results
- Patch to an open-source BBR code base
- Github pull request with draft text
- Pseudocode
- Github issue describing the idea
- CCWG email/meeting suggestion

Goals of evolving the BBR draft text

Goals as we evolve the BBR draft text:

- Clarification
- Simplification
- Better coexistence with Reno/CUBIC
- Better performance
- Avoiding performance regressions in the real world

Proposed bar for publication (keep in mind the target is an experimental RFC):

- Multiple deployments at scale in QUIC and TCP
- Text both TCP and QUIC implementations can follow
- Fair sharing with other BBR flows, coexistence with Reno and Cubic

Thesis: It's better to publish a good draft with deployment experience in a reasonable timeframe than evolve BBR indefinitely without shipping an RFC.