BBRv2 Update:
Internet Drafts & Deployment Inside Google

**TCP BBR:** Neal Cardwell, Yuchung Cheng, Kevin Yang
Soheil Hassas Yeganeh, Priyanjan Jha, Yousuk Seung, Luke Hsiao, Matt Mathis
Van Jacobson

**QUIC BBR:** Ian Swett, Bin Wu, Victor Vasiliev

[https://groups.google.com/d/forum/bbr-dev](https://groups.google.com/d/forum/bbr-dev)

IETF 112: Online; Nov 8, 2021
BBRv2 deployment status at Google: deployed for all internal production TCP traffic
- BBRv2 code status
- Overview of BBR Internet Drafts updated to describe BBRv2:
  - Delivery rate estimation draft
  - BBR congestion control draft

Target for this talk:
- Responding to requests from other transport stack maintainers implementing BBRv2
- Documenting the BBRv2 algorithm
- Inviting the community to:
  - Read the drafts and offer editorial feedback
  - Share algorithm or code fixes or enhancements
  - Share test results
  - Post bug reports
Update on BBR deployment status at Google

- Google-internal traffic uses either BBv2 (default) or BBRv2.Swift (in pilot):
  - BBv2 is current default TCP congestion control for internal TCP traffic
    - Using ECN, loss, bandwidth, RTT as signals
  - BBRv2.Swift is in pilot as next TCP congestion control for intra-datacenter use
    - Using network_RTT (excluding receiver delay) in addition to BBRv2 signals above
- Google-external traffic: YouTube, google.com transitioning to BBv2
  - BBv1 is current TCP and QUIC congestion control for external traffic
  - A/B experiments with BBv2 for a small percentage of users for TCP and QUIC
    - Reduced queuing delays: RTTs lower than BB v1 and CUBIC
    - Reduced packet loss: loss rates closer to CUBIC than BB v1
- Continuing to iterate using production experiments and lab tests
Status of BBR v2 Code

● TCP BBRv2 "alpha/preview" release:
  ○ Linux TCP (dual GPLv2/BSD): [github.com/google/bbr/blob/v2alpha/README.md](http://github.com/google/bbr/blob/v2alpha/README.md)
  ○ Recent updates:
    ■ Rebased to Linux v5.10
    ■ Various minor bug fixes
● QUIC BBR v2 "alpha/preview" release:
  ○ Chromium QUIC (BSD):on chromium.org in bbr2_sender.{ cc, h }
● BBR v2 alpha release is ready for research experiments; we invite:
  ■ Ideas for test cases and metrics to evaluate
  ■ Test results and traces
  ■ Algorithm/code ideas
  ■ Patches
● BBR v2 algorithm: IETF 104 [slides | video]; alpha code: IETF 105 [slides | video]
Delivery rate estimation Internet Draft is here:
  - draft-cheng-iccrg-delivery-rate-estimation

Covers bandwidth sampling mechanism used by BBRv1 and BBRv2
  - Largely unchanged since BBRv1
  - One significant fix in detecting application-limited rate samples:
    - A connection deciding to retransmit a packet is equivalent to an application writing data into a connection.
    - Bubbles of idleness can happen before either event.
  - Other minor fixes/edits
  - Overview and diagrams in IETF 99 slides still apply
BBR congestion control Internet Draft is here:
  ○ [draft-cardwell-iccrg-bbr-congestion-control](https://example.com)

Latest rev covers current BBRv2 algorithm
  ○ Includes aspects relevant to current public Internet:
    ■ Loss response
    ■ Strategy for coexistence with CUBIC/Reno
  ○ Does not (yet) include aspects for other environments (datacenter or L4S):
    ■ ECN response
    ■ Swift-style RTT response
  ○ Includes known issues that are a WIP:
    ■ Current algorithm is not aggressive enough in increasing inflight_hi after loss if there is a standing queue (causes CUBIC or incumbent BBRv2 flows to have higher throughput)

Presentation today will focus on:
  ○ An overview of the main contents of BBR Internet Draft
  ○ Graphical diagrams that would be hard to convey in ASCII art in Internet Draft
Terminology

Design overview
- Network path model
- Target operating point
- Control parameters
- State machine overview
- Algorithm organization

Detailed algorithm
- Maintaining the network path model
  - Rate (bandwidth)
  - Volume (in-flight data)
  - BDP
- Setting control parameters
  - Pacing rate
  - Send quantum
  - Congestion window

Detailed algorithm (cont.)
- State machine
  - Initialization
  - Startup
  - Drain
  - ProbeBW
    - ProbeBW_DOWN
    - ProbeBW_CRUISE
    - ProbeBW_REFILL
    - ProbeBW_UP
  - ProbeRTT
BBRv2 congestion control: block diagram

**Input signals:** throughput, RTT, loss, ECN

**Model of the Network Path**

**Probing State Machine**

- Increases / decreases sending rate w.r.t. target rate / volume

**Pacing Engine**

- Rate quantum
- cwnd

- **target inflight = est. BDP**
BBRv2 Network Path Model

Data rate:
- bw_lo → max_bw → bw_hi → min() → bw

Data volume:
- min_rtt
- extra_acked
- offload_budget → bdp
- max_inflight
- inflight_lo → min() → inflight_hi

Control outputs:
- cwnd
- pacing_rate → quantum
BBRv2 State Machine

- State machine for 2-phase sequential probing:
  - Accelerate to raise throughput, probe max bandwidth
  - Decelerate to reduce queuing, probe min RTT

Startup: double bw each round until flow estimates pipe is full
Drain: drain the estimated queue from the bottleneck
ProbeBW:DOWN: cut rate/inflight to drain queue, yield bw
ProbeBW:CRUISE: try to match sending rate to delivery rate
ProbeBW:REFILL: try to refill the "pipe" but not the queue
ProbeBW:UP: probe for available bw at the risk of queuing/loss
ProbeRTT: if needed, a coordinated inflight cut to probe RTT
BBRv2 State Machine: Example "Life of a Flow"

- **Startup**
- **Drain**
- **Cruise**
- **ProbeBW:REFILL**
- **ProbeBW:UP**
- **ProbeBW:DOWN**
- **Cruise**

**Volume of data in flight**

**In-flight**
- `inflight_hi`
- `bdp`
- `inflight_lo`

**Time**
Conclusion

● Actively working on BBRv2, BBRv2.Swift at Google for:
  ○ Linux TCP
  ○ Chromium QUIC
● BBR Internet Drafts have been updated to cover BBRv2:
  ○ Delivery rate estimation: draft-cheng-iccrq-delivery-rate-estimation
  ○ BBRv2 congestion control: draft-cardwell-iccrq-bbr-congestion-control
● We invite the community to share…
  ○ Feedback on the drafts
  ○ Test results, issues, patches, or ideas
● Thanks!
https://groups.google.com/d/forum/bbr-dev

Internet Drafts, paper, code, mailing list, talks, etc.

Special thanks to Eric Dumazet, Nandita Dukkipati, C. Stephen Gunn, Jana Iyengar, Pawel Jurczyk, Biren Roy, David Wetherall, Amin Vahdat, Leonidas Kontothanassis, and {YouTube, google.com, SRE, BWE} teams.