A Performance Evaluation of TCP BBRv2 Alpha

Jose Gomez*, Elie Kfoury*, Jorge Crichigno*, Elias Bou-Harb† and Gautam Srivastava‡
*Integrated Information Technology, University of South Carolina, Columbia, U.S.A
†The Cyber Center For Security and Analytics, University of Texas at San Antonio, San Antonio, U.S.A
‡Department of Mathematics and Computer Science, Brandon University, Canada
Email: {gomezgaj, ekfoury}@email.sc.edu, jcrichigno@cec.sc.edu, elias.bouharb@utsa.edu, srivastavag@brandonu.ca

***Abstract*—The alpha version of Bottleneck Bandwidth and Round-trip Time version 2 (BBRv2) has been recently presented, which aims to mitigate the shortcomings of its predecessor, BBR version 1 (BBRv1). Previous studies show that BBRv1 provides a high link utilization and low queuing delay by estimating the available bottleneck bandwidth. However, its aggressiveness induces unfairness when flows i) use different congestion control algorithms, such as CUBIC, and ii) have distinct round-trip times (RTTs). This paper presents an experimental evaluation of BBRv2, using Mininet. Results show that the coexistence between BBRv2-CUBIC is enhanced with respect to that of BBRv1-CUBIC, as measured by the fairness index. They also show that BBRv2 mitigates the RTT unfairness problem observed in BBRv1. Additionally, BBRv2 achieves a better fair share of the bandwidth than its predecessor when network conditions such as bandwidth and latency dynamically change. Results also indicate that the average flow completion time of concurrent flows is reduced when BBRv2 is used.**

***Keywords*—Bottleneck Bandwidth and Round-trip Time (BBR), congestion control, bandwidth-delay product (BDP), CUBIC, router buffer size, RTT unfairness.**

I. Introduction

The Transmission Control Protocol (TCP) [1] has been the standard transport protocol to establish a connection between end devices. As a robust, well-established protocol capable of providing data delivery in the face of packet losses, TCP has been the protocol of choice for decades. Moreover, an increased number of disciplines rely on high-speed networks to support reliable large data transfers. Examples include high-definition video conferencing systems [2] and global distribution of massive datasets [3], [4].

An essential feature of TCP is the congestion control, which is aimed to probe for the available capacity of the network to determine how many packets the sender can transmit safely. In the late 1980s, Jacobson et al. [5] described the principles of window-based congestion control algorithms. Thereafter, many improvements have been devised [6]. In particular, TCP uses an additive increase multiplicative decrease (AIMD) algorithm to establish the TCP sending rate. TCP linearly increases its congestion window size (and hence its transmission rate) until a triple duplicate-acknowledgement event occurs as a result of a packet loss. It then decreases its congestion window size by a factor of two.

This work was supported by the U.S. National Science Foundation, Office of Advanced Cyberinfrastructure, Award #1925484.

Router buffers are aimed to avoid losses by temporarily buffering packets as transitory bursts dissipate. If the router has a small buffer, packets may be dropped even in the absence of congestion. When bottleneck buffers are large, loss-based congestion control keeps them full, causing excessive delay or bufferbloat. When bottleneck buffers are small, loss-based congestion control misinterprets loss as a signal of congestion, leading to low throughput [7].

BBRv1 is loss agnostic and does not follow the AIMD rule. Instead, it actively estimates the bottleneck bandwidth and the RTT, which are then used to establish the sending rate [7]. Although BBRv1 produces higher throughput than traditional loss-based congestion control, it suffers from the RTT unfairness problem (BBRv1 allocates more bandwidth to flows with large RTTs) and poor coexistence with other algorithms [8].

BBRv2 aims to mitigate the limitations of BBRv1 [9]. BBRv2 is a hybrid congestion control algorithm that combines rate-based and model-based approaches. This means that the algorithm actively measures the bottleneck bandwidth, the RTT, and the packet loss rate to build a model of the end-to-end path. In contrast to BBRv1, which does not consider packet losses and explicit congestion notification (ECN) as inputs, BBRv2 uses these variables to estimate the bandwidth-delay product (BDP) and sending rate. Moreover, BBRv2 holds short-term and long-term inflight estimates of the bottleneck bandwidth and the inflight volume. This mechanism consists of a short-term slow start threshold estimate and long-term maximum congestion window [9].

This paper presents an experimental evaluation of BBRv2, using Mininet. It studies its throughput performance and explores the coexistence between BBRv2 and CUBIC flows and the RTT unfairness problem. The experiments consider scenarios that involve routers with different buffer sizes and variable RTTs. The rest of the paper is organized as follows: Section II presents the related works. Section III describes the experimental setup. Section IV presents the experimental results, and Section V concludes the paper.

II. Related Work

Previous studies on BBRv1 considered various network conditions to provide an in-depth analysis of its behavior. Scholz et al. [10] conducted experiments using BBRv1 and reported relevant results, such as bottleneck overestimation,

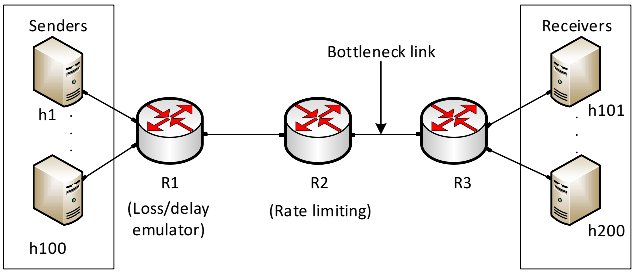
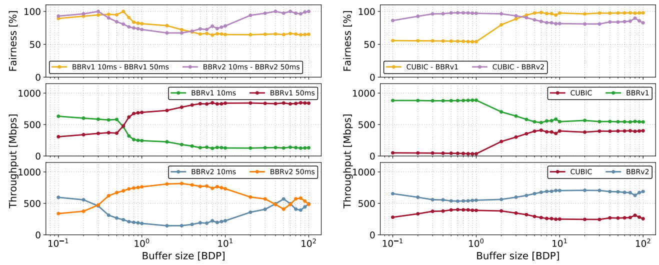


Fig. 1. Topology used for evaluations.



(a) RTT unfairness (b) CUBIC vs BBRv1/BBRv2

Fig. 2. Throughput and fairness index as functions of the buffer size, for 100 competing flows. (a) 50 flows have 10ms RTT and the other 50 flows have 50ms RTT. (b) 50 flows use CUBIC and the other 50 flows use BBRv1/BBRv2.

inter-protocol behavior with CUBIC and RTT unfairness. Hock et al. [8] showed that BBRv1 introduces a high packet retransmission rate. Zhang et al. [11] proposed an adapted version of BBRv1 called modest BBR. This approach primarily aims at reducing the packet retransmission rate and adjusting the pacing rate according to the network condition. The authors reported that modest BBR achieves high throughput and better inter-protocol fairness with loss-based schemes. Ma et al. [12] focused on the RTT unfairness problem and proposed a scheme that alleviates this problem.

Fejes et al. [13] conducted experimental evaluations to analyze the fairness and coexistence of TCP flows. The experiments combined different Active Queue Management (AQM) and congestion control algorithms, which have evolved independently during the last decades. The authors determined that such combinations result in poor fairness because ideal assumptions used during the isolated development of algorithms do not hold in a heterogeneous network. Kim et al. [14] proposed a variant of BBRv1 called delay-aware BBR (DA-BBR), which focuses on improving the RTT unfairness.

Bensley et al. [15] showed that when routers use ECN, BBRv2 achieves high-burst tolerance, low latency, and high throughput, even with small buffers. Kfoury et al. [16] proposed a novel scheme based on programmable switches.The scheme relies on a custom protocol embedded in the IP options header field, which is parsed by programmable P4 switches. With input from switches, end devices are dynamically notified to adjust the pacing rate. The scheme increases throughput and enhances fairness.

Previous work focused on a limited subset of network conditions when testing BBRv2 [17]. This work presents an experimental evaluation that entails different network conditions and a large number of flows. The main goal is to study the throughput performance of BBRv2 and its behavior regarding RTT and inter-protocol unfairness.

III. Experimental Setup

Consider Figure 1. The network used to conduct the experiments consists of 100 senders (h1, h2, ..., h100). Each sender opens a TCP connection to its corresponding receiver (h101, h102, ..., h200). The AQM policy used in routers is Tail Drop. The emulation is conducted in Mininet [18] which uses Linux network namespaces, a lightweight mechanism for isolating network resources. In the emulated environment, sufficient CPU cores ($\sim$1 CPU core per device, Xeon 6130 operating at 2.1 GHz) were allocated to avoid over-utilization of resources. The CPU usage was kept below prudent levels.

a) Loss/delay emulation: The router R1 is used to inject delay and packet losses on the link connected to router R2, in order to allow configurable RTTs and packet losses. The Network Emulator (NetEm) tool [19] is used to set the values of delay and packet loss rate.

b) Rate limitation and buffer size: Router R2 uses the Token Bucket Filter (TBF) to emulate a bottleneck by limiting the link rate. Additionally, TBF is used to set the buffer size on the egress interface of router R2 (the interface that connects to router R3). The bottleneck bandwidth (link R2-R3) is set to 1Gbps. All other links have a capacity of $\sim$40Gbps.

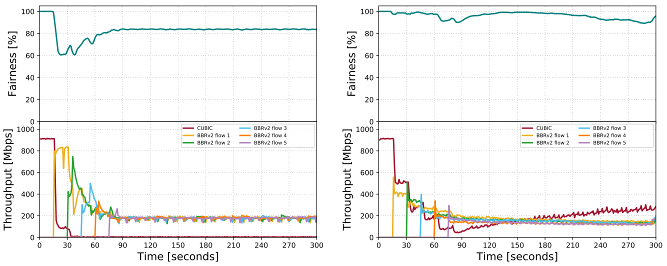
c) Metrics collection: The tool used to measure performance is iPerf3 [20]. Performance metrics and variables include throughput, fairness index, RTT, and flow completion time. The fairness index is reported and computed according to the RFC 5166 recommendation [21].

IV. Results and Evaluation

Given a network condition with specific parameters (e.g., buffer size, RTT), the experiment was repeated 10 times and the corresponding average is reported.

A. Round-trip Time Unfairness

Figure 2(a) reports the fairness index in percentage and the throughput of BBRv1 and BBRv2, as a function of the buffer size. There are 50 flows with a RTT of 10ms and 50 flows with a RTT of 50ms. Consider first the graphs for BBRv1. When the buffer size is small, the fairness index approaches 100%. However, when the buffer size increases above 0.6BDP, BBRv1 allocates more bandwidth to flows with 50ms RTT and the RTT unfairness is observed. Consider now the graphs for BBRv2. When the buffer size is small, the fairness index approaches 100%. When the buffer size is between 0.4-12 BDP, the RTT unfairness is noted. However, as the buffer size increases above 12BDP, the fairness index approaches 100% again. With large buffer sizes, BBRv2 produces fairer allocation than BBRv1.



(a) A CUBIC flow with five subsequent BBRv1 flows (b) A CUBIC flow with five subsequent BBRv2 flows

Fig. 3. Throughput and fairness index as a function of the time between a CUBIC flow and subsequently BBRv1 and BBRv2 flows. (a) A BBRv1 flow joins every 15 seconds (b) A BBRv2 flow joins every 15 seconds.

Figure 2(b) presents the throughput and fairness index as a function of the buffer size when CUBIC flows compete with BBRv1 flows and BBRv2 flows. Consider first the graphs for BBRv1. When the buffer size is below 1BDP, most of the bandwidth is allocated to BBRv1 flows and the fairness index is approximately 50%. As the buffer size increases above 1BDP, the bandwidth allocation is fairer and the fairness index approaches 100%. Consider now the graphs for BBRv2. When the buffer size is 1BDP or smaller, the fairness index approaches 100%. Although the fairness index then decreases slightly as the buffer size increases, its value is always above ∼80%. In summary, BBRv2 demonstrates a better coexistence with CUBIC than BBRv1 when the buffer size is small.

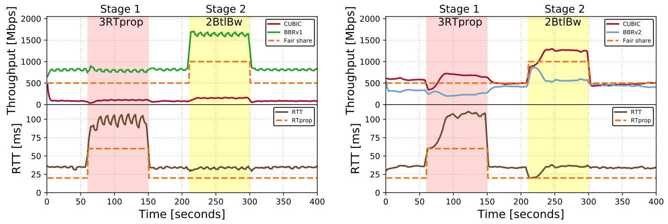
B. Accumulating Effects

This section evaluates the negative impact of BBR flows joining a CUBIC flow. The average throughput and the fairness index are presented as a function of time. All flows share a 1 Gbps bottleneck link and the RTT is 20ms.

Figure 3(a) shows a CUBIC flow at t_0 and five subsequent BBRv1 flows joining every 15 seconds. When the first BBRv1 flow joins the network, the throughput of the CUBIC flow rapidly decreases below 100 Mbps. The remaining bandwidth is consumed by the BBRv1 flow consequently, the fairness index establishes around ∼60%. When the second flow joins at t_{30}, the throughput of the CUBIC flow is completely consumed by BBRv1 flows and the fairness index is below 80%. When the third flow joins at t_{45}, the CUBIC flow maintains a very low throughput. After the fourth flow joins at t_{60}, the throughput of the CUBIC flow remains low and the fairness index does not achieve a value greater than 80%. At t_{75}, when the last flow joins, all the available bandwidth is shared by the BBRv1 flows, the CUBIC flow suffers low throughput and the fairness index does not increase more than ∼80%.

Figure 3(b) depicts a scenario for a CUBIC flow and five subsequent BBRv2 flows joining every 15 seconds. At t_{15}, it is observed that the throughput of the CUBIC flow is not absorbed by the incoming BBRv2 flow. When the third BBRv2 flow joins at t_{45}, the CUBIC flow converges to a fair share and the fairness index does not go below 90%. At t_{60}, the fourth flow joins and there is appreciated an impact on the performance of the CUBIC flow. It is also observed that the fairness index does not decrease below 80%. Similar behavior is observed when the fifth flow joins at t_{75}. After all the flows join, the fairness index does not decrease below ∼80% and the final value settles around 100%.

In summary, results show that BBRv2 flows are less aggressive than BBRv1 flows when they are sharing the same bottleneck link with a CUBIC flow. Moreover, BBRv2 flows are capable to maintain a high fairness index and, therefore, a better coexistence with a CUBIC flow compared to BBRv1.



(a) CUBIC and BBRv1 flows (b) CUBIC and BBRv2 flows

Fig. 4. Two TCP flows under changing network conditions. (a) CUBIC and BBRv1 flows. (b) CUBIC and BBRv2 flows.

C. Changing network conditions

The following experiment presents a scenario where coexisting CUBIC and BBR flows react to changes in network conditions. The experiment evaluates the throughput of both flows and the RTT of the bottleneck link under two network changes specified as stages. Before the first stage starts, the bottleneck bandwidth (BtlBw) is 1 Gbps and the round-trip time (RTprop) propagation is 20ms. In the first stage (highlighted with red), the round-trip time propagation at the bottleneck link increases from 20ms to 60ms (3RTprop) for 90 seconds. After that, the network returns to the previous condition. Then, the second stage ((highlighted with yellow)) starts by increasing the bottleneck bandwidth from 1Gbps to 2Gbps (2BtlBw). The duration of the second stage is 90 seconds. Then, the network condition returns to the default configuration (i.e. 1Gbps BtlBw, 20ms RTprop).

Figure 4(a) shows a scenario where a CUBIC and BBRv1 start sending data at the same time. It is observed that independently of the changes in the network conditions, BBRv1 utilizes the major part of the available bandwidth degrading CUBIC performance. In stage 1, when the RTprop has tripled the link experiences an increase from ∼60ms to ∼100ms in the RTT. In stage 2, it is observed that BBRv1 immediately utilizes the major part of the bandwidth consequently, both flows are not able to reach a fair share.

Figure 4(b) depicts a scenario where a CUBIC and BBRv2 flows share the same bottleneck link. The increase in the RTprop in stage 1 leads to RTT values over 100ms. In stage 2, the CUBIC flow is occupying more bandwidth than BBRv2. After restoring the default network condition (∼300 seconds), CUBIC and BBRv2 flows converge to a fair share.

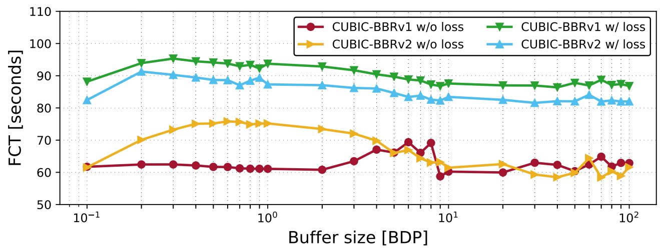


Fig. 5. Average flow completion time of 100 flows as a function of the buffer size. The four curves represent the results of flows operating according to the following configurations: 50 CUBIC flows and 50 BBRv1 flows without packet losses (red); 50 CUBIC flows and 50 BBRv2 flows without packet losses (yellow); 50 CUBIC flows and 50 BBRv1 flows with 1% emulated packet loss (green); 50 CUBIC flows and 50 BBRv2 flows with 1% emulated packet loss (blue).

In summary, BBRv2 is closer to a fair share than BBRv1 under changing network conditions thus, BBRv2 presents a better coexistence with a CUBIC flow.

D. Flow Completion Time

Lastly, Figure 5 reports the average flow completion time (FCT) of 100 competing flows (50 CUBIC and 50 BBRv1/BBRv2). In this experiment, the average FCT is presented as a function of the buffer size. Two scenarios are considered: 1) without packet losses, 2) with emulated packet losses. In a scenario without packet losses, it is observed that CUBIC-BBRv1 flows present a lower completion time than CUBIC-BBRv2 flows when the buffer size is smaller than 3BDP. For buffer sizes greater than 3BDP, CUBIC-BBRv1/BBRv2 flows present a completion time of around 60 seconds. However, in the presence of a packet loss rate of 1%, CUBIC-BBRv2 flows present a completion time from 2% to 4% lower than flows with CUBIC-BBRv1 for all buffer sizes

V. Conclusion

BBRv1 represented a significant disruption to the traditional congestion control algorithm as it is not driven by the AIMD control law (i.e., packet losses are not used as a signal to modify the sending rate). Despite its success in improving the throughput, BBRv1 presented some issues, including the poor coexistence with traditional congestion control algorithms such as CUBIC. In this context, BBRv2 has been proposed to address such issues.

This work presents an experimental evaluation of BBRv2 using Mininet. Results show that BBRv2 presents a better coexistence with CUBIC flows with respect to its predecessor, BBRv1. It is also reported that BBRv2 mitigates the RTT unfairness problem observed in BBRv1. Moreover, BBRv2 is capable to achieve a better fair share of the bandwidth compared to BBRv1 when network conditions such as bandwidth and latency dynamically change. Finally, results also indicate that the average flow completion time of concurrent flows is reduced when BBRv2 is used in the presence of packet loss.

References

[1] J. Postel, “Rfc0793: Transmission control protocol,” 1981.
[2] E. Kfoury, J. Crichigno, and E. Bou-Harb, “Offloading media traffic to programmable data plane switches,” in *IEEE International Conference on Communications (ICC)*, IEEE, June 2020.
[3] N. Hanford, B. Tierney, and D. Ghosal, “Optimizing data transfer nodes using packet pacing,” in *Proceedings of the Second Workshop on Innovating the Network for Data-Intensive Science*, p. 4, ACM, 2015.
[4] J. Crichigno, E. Bou-Harb, and N. Ghani, “A comprehensive tutorial on science dmz,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 2041–2078, 2019.
[5] V. Jacobson, “Congestion avoidance and control,” *ACM SIGCOMM computer communication review*, vol. 18, no. 4, pp. 314–329, 1988.
[6] R. Al-Saadi, G. Armitage, J. But, and P. Branch, “A survey of delay-based and hybrid tcp congestion control algorithms,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3609–3638, 2019.
[7] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, “Bbr: Congestion-based congestion control,” *Queue*, vol. 14, no. 5, pp. 20–53, 2016.
[8] M. Hock, R. Bless, and M. Zitterbart, “Experimental evaluation of bbr congestion control,” in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, pp. 1–10, IEEE, 2017.
[9] N. Cardwell, Y. Cheng, S. H. Yeganeh, I. Swett, V. Vasiliev, P. Jha, Y. Seung, M. Mathis, and V. Jacobson, “Bbrv2: A model-based congestion control,” in *Presentation in ICCRG at IETF 104th meeting*, 2019.
[10] D. Scholz, B. Jaeger, L. Schwaighofer, D. Raumer, F. Geyer, and G. Carle, “Towards a deeper understanding of tcp bbr congestion control,” in *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, pp. 1–9, IEEE, 2018.
[11] Y. Zhang, L. Cui, and F. P. Tso, “Modest bbr: Enabling better fairness for bbr congestion control,” in *2018 IEEE Symposium on Computers and Communications (ISCC)*, pp. 00646–00651, IEEE, 2018.
[12] S. Ma, J. Jiang, W. Wang, and B. Li, “Towards rtt fairness of congestion-based congestion control,” *CoRR*, 2017.
[13] F. Fejes, G. Gombos, S. Laki, and S. Nádas, “Who will save the internet from the congestion control revolution?,” in *Proceedings of the 2019 Workshop on Buffer Sizing*, pp. 1–6, 2019.
[14] G.-H. Kim and Y.-Z. Cho, “Delay-aware bbr congestion control algorithm for rtt fairness improvement,” *IEEE Access*, 2019.
[15] S. Bensley, L. Eggert, D. Thaler, P. Balasubramanian, and G. Judd, “Datacenter tcp (dctcp): Tcp congestion control for datacenters,” *Internet Draft*, 2017.
[16] E. F. Kfoury, J. Crichigno, E. Bou-Harb, D. Khoury, and G. Srivastava, “Enabling tcp pacing using programmable data plane switches,” in *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, pp. 273–277, IEEE, 2019.
[17] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, I. Swett, J. Iyengar, V. Vasilev, and V. Jacobson, “Bbr congestion control: Ietf 100 update: Bbr in shallow buffers,” in *Proc. IETF-100*, 2017.
[18] R. L. S. De Oliveira, C. M. Schweitzer, A. A. Shinoda, and L. R. Prete, “Using mininet for emulation and prototyping software-defined networks,” in *2014 IEEE Colombian Conference on Communications and Computing (COLCOM)*, pp. 1–6, IEEE, 2014.
[19] S. Hemminger *et al.*, “Network emulation with netem,” in *Linux conf au*, pp. 18–23, 2005.
[20] J. Dugan, S. Elliott, B. A. Mah, J. Poskanzer, and K. Prabhu, “iperf3, tool for active measurements of the maximum achievable bandwidth on ip networks,” *URL: https://github. com/esnet/iperf*.
[21] S. Floyd, “Metrics for the evaluation of congestion control mechanisms”, rfc 5166,” 2008.