Understanding the Performance of TCP BBRv2 using FABRIC

Jose Gomez*, Elie Kfoury*, Jorge Crichigno*, Gautam Srivastava†
*Integrated Information Technology Department, University of South Carolina, Columbia, SC, USA
†Department of Mathematics and Computer Science, Brandon University, Canada
{gomezgaj, ekfoury}@email.sc.edu, jcrichigno@cec.sc.edu, srivastavag@brandonu.ca

***Abstract*—This paper presents a performance evaluation of the Bottleneck Bandwidth and Round-trip Time version 2 (BBRv2) TCP congestion control. The experiments are conducted in FABRIC, a national-scale experimental network infrastructure that enables large-scale testing. Google released BBRv2 in 2019 as an improvement over its predecessor, BBRv1. Previous evaluations showed that BBRv2 demonstrates better coexistence with loss-based congestion control algorithms (CCAs), presents low retransmission rates, and produces shorter queueing delays even with large buffers. Evaluations conducted in this paper used FABRIC to reproduce the network conditions observed in Wide Area Networks (WANs). The tests presented in this paper evaluate the throughput as a function of the Round-trip Time (RTT) of BBRv2 compared to various CCAs, the RTT unfairness of BBRv1 and BBRv2, the queue occupancy, and the packet loss rate as a function of the router's buffer size. Additionally, this paper presents and discusses the influence of Active Queue Management (AQM) algorithms to mitigate performance degradation produced by the RTT unfairness and the interaction of different CCAs when sharing a bottleneck link.**

***Index Terms*—Bottleneck Bandwidth and Round-trip Time (BBR), congestion control, Bandwidth-Delay Product (BDP), router's buffer size, RTT unfairness, FABRIC.**

I. Introduction

The Transmission Control Protocol (TCP) [1] is the most widely adopted transport protocol that enables reliable end-to-end data transfers between applications. A key function of TCP is its congestion control mechanism, which regulates the sending rate in the presence of congestion. Various TCP congestion control implementations aim to mitigate congestion by reacting to packet loss, delay, and other network metrics. Early TCP variants considered packet losses as a signal of congestion [2], other implementations considered delay as an indication of network congestion [3], whereas hybrid approaches used the combination of multiple metrics [4]. The Additive Increase Multiplicative Decrease (AIMD) control principle governs traditional TCP CCAs. Such algorithms include CUBIC, Reno, HTCP, and others. More recently, BBRv1 periodically estimates the bottleneck bandwidth and uses pacing to set the sending rate. This approach differs from traditional loss-based CCAs.

Although BBRv1 improved TCP throughput, it presents an unfair bandwidth share with other TCP flows and high retransmission rates. BBRv2 has been proposed as a solution to the shortcomings of BBRv1. The approach of BBRv2 is to create a network traffic model by estimating the bandwidth, observing the RTT, measuring the loss rate, and incorporating Explicit Congestion Notification (ECN) capabilities. The literature [5–7] reported that BBRv2 tolerates higher packet loss rates than loss-based CCAs, presents lower retransmission rates than BBRv1, shows a better coexistence with CUBIC, reduces the impact of the RTT unfairness, and exhibits lower queueing delays.

While previous studies have made significant contributions to understanding the performance of BBRv2 using emulation tools and real hardware, there is a gap in the literature. Specifically, there is a need for a performance evaluation that utilizes large-scale testbeds. This paper aims to fill this gap by presenting an experimental evaluation of BBRv2 using FABRIC [8], a novel research infrastructure that supports large-scale research in various domains such as networking, distributed computing, machine learning, and science applications. By leveraging large-scale testbeds, this paper evaluates the behavior of BBRv2 with realistic network conditions. The contributions of this paper can be summarized as follows:

- Evaluating the performance of BBRv2 using WAN conditions.
- Illustrating the influence of FABRIC's inherent delay on BBRv1 and BBRv2 bottleneck bandwidth estimation.
- Demonstrating that BBRv2 attains comparable performance to BBRv1 presenting a lower retransmission rate.
- Analyzing the impact of different buffer sizes, propagation delays, and loss rates on the performance of BBRv2.

The experiments presented in this paper consider scenarios that involve TCP flows with different RTTs, routers with small and large buffer sizes, and queues controlled with AQMs. The rest of the paper is organized as follows: Section II presents the related works. Section III explains the motivation of this work. Section IV provides background on BBRv2 and FABRIC. Section V describes the experiments and the results. Section VI discusses the limitations and section VII concludes the paper.

II. Related Work

Kfoury *et al.* [5] used Mininet to conduct a performance evaluation of BBRv2. Mininet [9] is a network emulator based on namespaces to run experiments using a real protocol stack in Linux. In this paper, the authors evaluated the alpha version

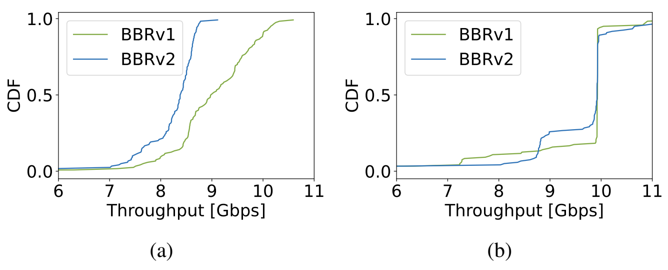


FIG. 1: CDFs of the bottleneck bandwidth estimation of BBRv1 and BBRv2. (a) with 45ms emulated delay. (b) with 45ms propagation delay.

of BBRv2 by conducting experiments with small and large buffer sizes, reproducing RTT unfairness scenarios, measuring the coexistence of BBRv2 with competing TCP flows, and evaluating the impact of AQMs in the performance of BBRv1 and BBRv2. Similarly, Gomez *et al.* [6] evaluated BBRv2 using a Mininet emulation. The authors demonstrated a better coexistence between BBRv2 and CUBIC than between BBRv1 and CUBIC. The paper shows how BBRv2 mitigates the RTT unfairness problem and presents a better bandwidth share than BBRv1 in changing network conditions.

Song *et al.* [7] evaluated and compared BBRv1 and BBRv2 using Mininet and a physical testbed. The authors showed how BBRv2 could alleviate the unfairness issues of BBRv1. In their evaluations, BBRv2 presented better fairness with competing flows in routers with small buffers. The paper demonstrated that BBRv1 flows experience flow synchronization and co-existence issues with loss-based CCAs. Tierney *et al.* [10] described and performed experiments to assess the suitability of BBRv2 for use on Data Transfer Nodes (DTNs). The authors tested BBRv2 in a production network and a controlled testbed environment. Their evaluations showed that BBRv2 presents better performance in large data transfer scenarios. The paper also highlights that BBRv2 is a promising option in high-speed short-queue networking environments. Moreover, the authors verified that results obtained with Mininet are valid on real networks.

Scherrer *et al.* [11] presented a fluid model of BBRv1 and BBRv2 to complement the previous studies. The authors conducted simulations using ns-3 under various network settings and presented analytical evaluations such as stability analysis. Their evaluations show that the proposed model can accurately predict BBRv2 dynamics. The paper also confirmed that BBRv2 mitigated the undesirable behavior of BBRv1 and identified the scenarios in which BBRv2 leads to bufferbloat and unfairness.

III. MOTIVATION

Discrete-event simulators are valuable tools for evaluating and testing network protocols, services, and applications [12, 13]. Similarly, network emulators enable researchers to test services and applications using actual protocol stacks [9]. However, they may face limitations in reproducing the behavior of large-scale and high-speed networks due to constraints in transmission speeds, processing power, and memory. On the other hand, network testbeds allow researchers to test real-world network conditions more accurately than emulators and simulators, as they use actual hardware infrastructure.

Fig. 1 shows the Cumulative Distribution Function (CDF) of the bottleneck bandwidth estimation performed by BBRv1 and BBRv2 in a 10Gbps bottleneck link with a 45ms delay. Both CCAs compute the RTT to estimate the available bandwidth [14, 15]. Therefore, the bottleneck bandwidth estimation is sensitive to RTT variations. Fig. 1(a) shows the bottleneck bandwidth estimation obtained in an emulated environment, whereas Fig. 1(b) shows the results obtained with a 45ms propagation delay resulting from the physical separation of FABRIC nodes. It is observed that an emulated environment induces more underestimations of the current bandwidth (i.e., 10Gbps) than the one with a real propagation delay. In this paper, we leverage the distributed architecture of the FABRIC testbed to reproduce WAN conditions and test the performance of BBRv2.

IV. BACKGROUND

A. FABRIC

FABRIC (Adaptive Programmable Research Infrastructure for Computer Science and Science Applications) is a novel research infrastructure aimed to support large-scale research in networking, distributed computing, machine learning, and science applications [8]. Its main goal is to provide an experimentation testbed to explore methods and techniques to overcome the current architectural limitations of the Internet. One of these limitations is produced by the middleboxes in the network, which breaks the Internet architectural principles and complicates the process of troubleshooting connection problems and testing novel ideas. FABRIC's architecture consists of distributed resources along national labs, campuses, and commercial collocation spaces. Each FABRIC site provides a large amount of computing (i.e., CPUs, GPUs, and FPGAs) and storage interconnected by high-speed, dedicated optical links. Additionally, FABRIC integrates specialized testbeds in areas such as 5G, IoT, and cloud computing to create a rich environment for a wide range of experiments.

FABRIC allows experimenters to create networks using VMs residing in different locations in the United States and, more recently, in Europe. This capability allows the distribution of computing power across different sites. Moreover, the testbed facilitates orchestrating regular tests by providing a bastion host that supports configuration scripts where the experimenter can reserve resources and run tests. These scripts can be shared and modified by other researchers to address their experimentation requirements.

B. BBRv2

BBRv2 [15] is a rate-based, model-based CCA created to overcome the shortcomings of BBRv1. BBRv2 measures the bandwidth, the RTT, the packet loss rate, and the ECN mark rate to estimate bottleneck bandwidth and to model the end-to-end path across the network, referred to as the network path model. Following this approach, experimental evaluations [5,

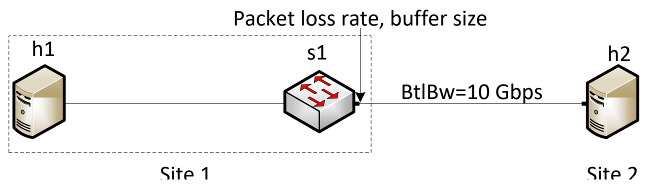


FIG. 2: Topology implemented in FABRIC to reproduce various RTTs. A 0.0046% loss rate is emulated using NetEm.

6] reported that BBRv2 maintains the high throughput and the bounded queueing delay properties of BBRv1. Additionally, BBRv2 tolerates much higher random packet loss rates than loss-based algorithms. Performance evaluations also show that BBRv2 has better coexistence with loss-based CCAs and lower retransmission rates than BBRv1. BBRv2 also presents a better fairness index with flows experiencing RTT unfairness [16].

This paper presents the results obtained with the alpha version of BBRv2 (v2alpha-2019-07-28) [17] running on FABRIC.

V. RESULTS AND EVALUATIONS

The following experiments use FABRIC to test BBRv2 against traditional CCAs such as CUBIC, Reno, HTCP, and its predecessor, BBRv1.

A. Experiment 1: Performance in a WAN with packet losses

Packet losses affect the performance of TCP and can occur at different devices in the network, including routers, switches, firewalls, and other network appliances. In the presence of packet losses, TCP CCAs such as Reno, CUBIC, and HTCP reduce their sending rate leading to lower performance. This issue increases the higher the RTT. On the other hand, BBRv1 follows a different approach and does not consider packet losses as a congestion indicator. Instead, BBRv1 periodically estimates the bottleneck bandwidth and uses pacing to set the sending rate to the estimated bottleneck bandwidth [14]. The literature reported that this approach leads to unfairness with other CCAs and a high retransmission rate [5, 7]. BBRv2 reduces the impact of this issue by considering packet losses as part of the bottleneck bandwidth estimation.

In this experiment, multiple FABRIC sites are used to reproduce different RTTs and observe the performance of data transfers using various CCAs. The goal of this experiment is to compare how a small packet loss rate affects the performance of CCAs such as CUBIC, Reno, HTCP, BBRv1, and BBRv2. Fig. 2 shows the topology used to conduct the experiments. Multiple sites are combined to produce the desired propagation delay.

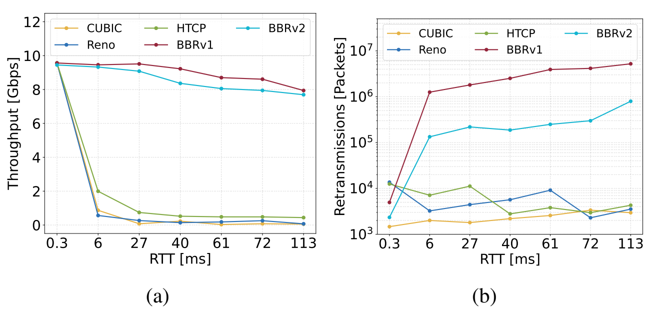


FIG. 3: Performance of CUBIC, Reno, HTCP, BBRv1, and BBRv2 as a function of the RTT. (a) Throughput. (b) Retransmissions.

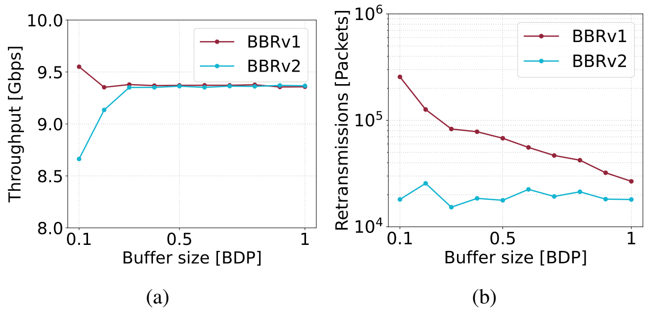


FIG. 4: Performance test as a function of the BDP. (a) Throughput. (b) Retransmissions.

The devices include three VMs implement the topology comprising two hosts (i.e., host h1 and host h2) and a switch (i.e., switch s1). The data transfers consist of a 120-second iPerf3 [18] test between a sender and a receiver. Experiments are repeated ten times, and the results are averaged. The bandwidth is limited using Token Bucket Filter (TBF), and packet losses are induced using NetEm [19].

Fig. 3 shows the TCP throughput of a data transfer across a 10Gbps path with different CCAs. The emulated packet loss rate is 0.0046% (i.e., 1/22,000). Fig. 3(a) shows that BBRv1 and BBRv2 perform between 8Gbps and ~10Gpbs for all RTTs, whereas the throughput of CUBIC, Reno, and HTCP collapses for RTTs greater than 6 milliseconds. Fig. 3(b) shows that the number of retransmissions produced by BBRv2 is around one million packets less than BBRv1, while BBRv2 achieves similar performance.

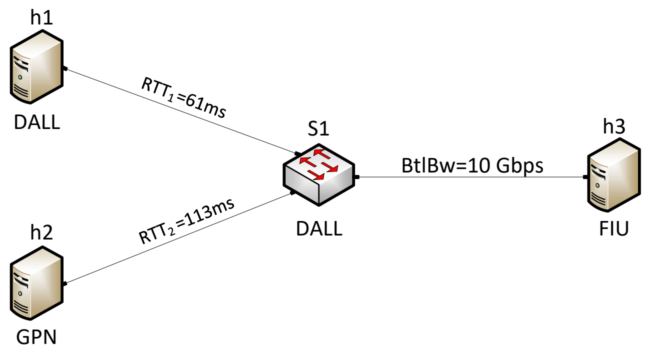
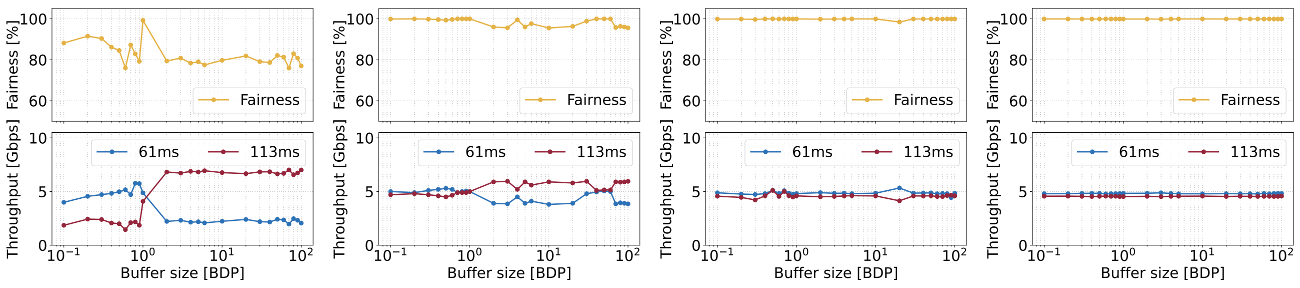


FIG. 5: Topology implemented in FABRIC to reproduce the RTT unfairness scenario. The RTT between h1 and h3 is 61ms (RTT_1), whereas the RTT between h2 and h3 is 113ms (RTT_2). The bottleneck bandwidth (BtlBw) is limited to 10Gbps using TBF.



(a) BBRv1 with Tail Drop (b) BBRv2 with Tail Drop (c) BBRv1 with FQ-CoDel (d) BBRv2 with FQ-CoDel

FIG. 6: Fairness index and throughput as functions of the buffer size for two competing flows: one flow has 61ms RTT, and the other flow has 113ms RTT. (a) Flows use BBRv1, and switch s1 implements a simple Tail Drop policy. (b) Flows use BBRv2, and switch S1 implements a simple Tail Drop policy. (c) Flows use BBRv1, and switch s1 implements FQ_CoDel. (d) Flows use BBRv2, and switch s1 implements FQ_CoDel.

B. Experiment 2: Retransmissions as a function of the buffer size

BBRv1 achieves high throughput and low latency by approximating the inflight data to the Badnwidth-Delay Product (BDP), which is the product of the bottleneck link capacity and the link's round trip delay [20]. However, its aggressiveness affects loss-based CCAs such as CUBIC, Reno, and HTCP, leading to massive packet retransmissions [21, 22]. BBRv2 mitigates this issue, presenting lower retransmissions and reaching similar throughput as BBRv1 [5].

This experiment evaluates the number of retransmissions as a function of the buffer size using BBRv1 and BBRv2. Fig. 2 shows the topology used to run this experiment. The RTT between host h1 and host h2 is 45 milliseconds, and the bandwidth was limited to 10Gbps using TBF. There are no emulated packet losses. The TCP flow is generated by running an iPerf3 test for 120 seconds. The measurements corresponding to each buffer size are repeated ten times and then averaged. Fig. 4 shows the throughput and retransmissions of BBRv1 and BBRv2 as a function of the buffer size from 0.1BDP to BDP. It is observed that the smaller the buffer size, BBRv1 shows a significantly higher number of retransmissions than BBRv2. On the other hand, BBRv2 presents similar throughput to BBRv1 for BDPs ranging from 0.3BDP to BDP while keeping the number of retransmissions at lower levels.

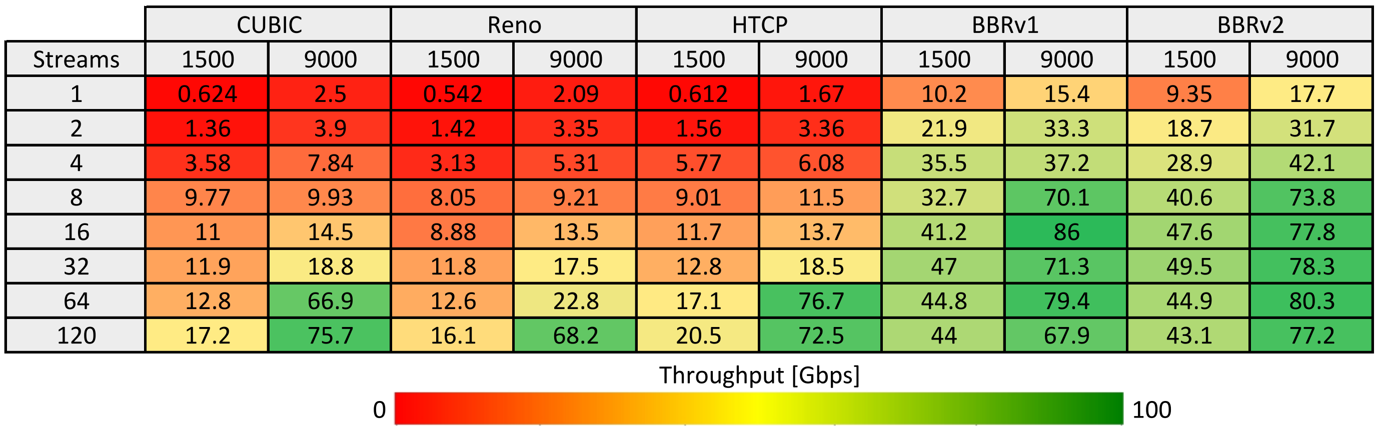
C. Experiment 3: RTT unfairness

In loss-based CCAs, the RTT unfairness occurs when transfers with smaller RTTs obtain a higher share of the bottleneck bandwidth [16]. This issue results from the probing frequency of flows with smaller RTTs that can recover faster after a loss event than flows with higher RTTs. Thus, two senders using the same bottleneck link will experience different throughputs if one is further away from the receiver than the other sender. On the other hand, BBRv1 does not consider packet losses and probes the network as a function of the RTT to estimate the bottleneck bandwidth. The higher the RTT, the more data is injected into the network. Therefore, flows with higher RTT are allocated with a higher bandwidth share. The literature has reported that BBRv1 suffers from RTT Unfairness [5, 6, 23] and BBRv2 reduces the impact of the RTT unfairness by considering packet losses when creating the model to estimate the bottleneck bandwidth.

In this experiment, two senders in different geographical locations initiate data transfers with a receiver to reproduce a scenario with RTT unfairness. The topology in Fig. 5 is implemented using a FABRIC slice comprising two senders (h1, h2), a switch (s1), and a receiver (h3). The senders are located in Dallas, Texas (DALL) and Columbia, Missouri (Great Plains Networks (GPN)), whereas the receiver is in Miami, Florida (Florida International University (FIU)). The VM used to reproduce a switch limits the rate and modifies the buffer size using TBF. The tests are orchestrated using Application Programming Interfaces (APIs) available in FABlib 1.4 [24]. The data transfers were conducted using iPerf3, running for 120 seconds. The tests are executed ten times, and the average is reported.

This experiment compares the RTT unfairness of BBRv1 and BBRv2 to observe whether BBRv2 mitigates this limitation. The first TCP flow occurs between DALL and FIU (RTT=61ms), whereas the second one is between GPN and FIU (RTT=113ms). The fairness and throughput are reported as a function of the buffer size.

Fig. 6(a) shows the throughput of BBRv1 flows. When the buffer size is below 1BDP, the fairness index is less than 90%. When the buffer size increases above 1BDP, the flow with 113ms RTT receives more bandwidth than the flow with 61ms RTT, resulting in a fairness index of around 80%. Fig. 6(b) shows the throughput of BBRv2 flows. The two flows receive similar amounts of bandwidth, and the fairness index remains above 85%. Figs. 6(c) and 6(d) show that by using FQ-CoDel, the RTT unfairness is eliminated and the fairness index is approximately 100% for both BBRv1 and BBRv2. Results also show that BBRv2 enhances the coexistence of flows with different RTTs, reducing the severity of the RTT unfairness. Additionally, implementing an AQM such as FQ-CoDel ensures a better coexistence between competing flows.



	CUBIC		Reno		HTCP		BBRv1		BBRv2	
Streams	1500	9000	1500	9000	1500	9000	1500	9000	1500	9000
1	0.624	2.5	0.542	2.09	0.612	1.67	10.2	15.4	9.35	17.7
2	1.36	3.9	1.42	3.35	1.56	3.36	21.9	33.3	18.7	31.7
4	3.58	7.84	3.13	5.31	5.77	6.08	35.5	37.2	28.9	42.1
8	9.77	9.93	8.05	9.21	9.01	11.5	32.7	70.1	40.6	73.8
16	11	14.5	8.88	13.5	11.7	13.7	41.2	86	47.6	77.8
32	11.9	18.8	11.8	17.5	12.8	18.5	47	71.3	49.5	78.3
64	12.8	66.9	12.6	22.8	17.1	76.7	44.8	79.4	44.9	80.3
120	17.2	75.7	16.1	68.2	20.5	72.5	44	67.9	43.1	77.2

FIG. 7: Average throughput belonging to different CCAs. The throughputs are given as a function of the number of streams and the MTU.

D. Experiment 4: Parallel streams and different MTUs

This experiment evaluates the maximum throughput that can be achieved between two sites in FABRIC using various CCAs. The topology used to run this experiment consists of two hosts (e.g., host h1 and host h2) connected via a layer two site-to-site network service [25]. The maximum link bandwidth is 100Gbps, and the latency between the sites is 26 milliseconds. This experiment measures TCP performance using iPerf with parallel streams. The end hosts are configured following the tuning guidelines in [26] but without pacing [27]. The hosts' interfaces are configured with 1500 bytes and 9000 bytes MTUs with Path MTU Discovery (PMTUD) enabled `net.ipv4.tcp_mtu_probing=1`. The latter configuration prevents PMTU black holes and allows TCP flows to achieve higher performance.

Fig. 7 shows the average throughput as a function of the number of parallel streams and the MTU. It is observed that higher performance is obtained with 9000 bytes MTUs. BBRv1 and BBRv2 achieve throughputs over 70Gbps with eight flows, whereas CUBIC, Reno, and HTCP need more than 64 flows to reach similar performance. With 1500 bytes MTUs, the maximum throughput is obtained with BBRv2 using 32 parallel streams.

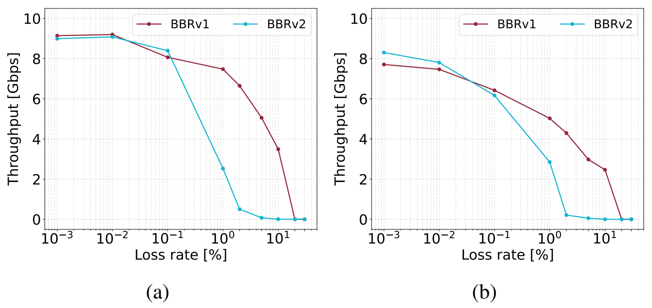


FIG. 8: Throughput as a function of packet losses. (a) RTT=26ms. (b) RTT=57ms.

E. Experiment 5: Throughput as a function of packet losses

This experiment measures the throughput of BBRv1 and BBRv2 as a function of the packet losses in scenarios with 26ms (DALL, SALT) and 52ms (UCSD, UMASS) propagation delays. Fig. 2 shows the topology used to run this experiment. Fig. 8 shows the test results where it is observed that BBRv1 achieves higher throughput with packet loss rates greater than 1%. On the other hand, the performance of BBRv2 is similar to the one of BBRv1 for packet loss rates lower than 1%.

F. Experiment 6: Queue occupancy

This experiment evaluates the queue occupancy in a router resulting from CUBIC, BBRv1, and BBRv2. The aim of this experiment is to observe the impact of enqueued packets on the performance of TCP flows. Routers implement queues with buffers, which are intended to absorb traffic bursts and reduce packet losses. However, there is no consensus on the right buffer size, which depends on metrics such as the link bandwidth, RTT, and the number of flows [28]. Traditional CCAs experience the bufferbloat problem [29]. On the other hand, BBRv1 and BBRv2 aim at keeping the queueing delay at lower levels.

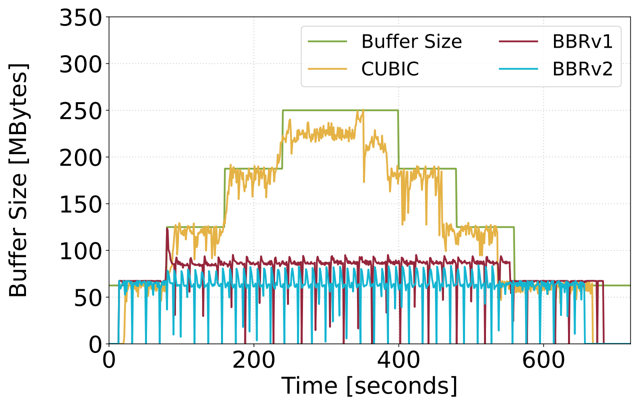


FIG. 9: Queue occupancy for different buffer sizes for CUBIC, BBRv1, and BBRv2.

The tests measure the queue occupancy during a data transfer over a 10Gbps link experiencing a 50ms delay. Fig. 2 presents the topology used to run this experiment. The initial buffer size is 1BDP (i.e., $\sim$59MBytes), which increases 1BDP every 80 seconds up to 4BDP. Then, the buffer sizes decrease to 1BDP. Results observed in Fig. 9 show the queue occupancy for CUBIC, BBRv1, and BBRv2 under changing buffer sizes. It is observed that CUBIC fills up the queue resulting in a higher end-to-end latency than BBRv1 and BBRv2.

VI. Limitations

Experimenters in FABRIC cannot configure intermediary devices such as routers and switches to modify parameters such as the buffer size, the number of queues, and the transmission rate. Therefore, experimenters must emulate such devices using a VM that acts as a switch or router. This approach produces results with lower realism than the ones that use real hardware. Furthermore, it should be noted that the Virtual Machines (VMs) utilized for conducting the experiments share Network Interface Cards (NICs) with the VMs reserved by other experimenters. While the networks themselves are isolated, it is possible for performance to be affected when two experimenters are simultaneously running tests.

VII. Conclusion

This paper evaluated the performance of BBRv2 using FABRIC to reproduce WAN conditions. Results show that BBRv2 performs similarly to its predecessor, BBRv1, presenting a lower retransmission rate. Results show that BBRv2 enhances the coexistence of flows with different RTTs using tail drop, the default queue management mechanism most routers use. AQMs such as FQ-CoDel mitigate the RTT unfairness issue by allocating flows into different queues and controlling the delay of enqueued packets. Moreover, higher throughputs can be achieved by tuning the network interfaces to handle packets with 9000 bytes MTUs. Finally, results show that the queue delay is lower when using BBRv1 and BBRv2 than when using traditional congestion control algorithms. Future works can evaluate the performance metrics using a P4-programmable switch to perform fine-grained measurements and compare them with the ones obtained with legacy devices.

VIII. Acknowledgement

The U.S. National Science Foundation supported this work under grant number 2118311.

References

[1] J. Postel, "Transmission control protocol," tech. rep., 1981.
[2] V. Jacobson, "Congestion avoidance and control," *ACM SIGCOMM Computer Communication Review*, 1988.
[3] 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*, 2019.
[4] J. Hespanha, S. Bohacek, K. Obraczka, and J. Lee, "Hybrid modeling of TCP congestion control," in *International Workshop on Hybrid Systems: Computation and Control*, 2001.
[5] E. Kfoury, J. Gomez, J. Crichigno, and E. Bou-Harb, "An emulation-based evaluation of TCP BBRv2 alpha for wired broadband," *Computer Communications*, 2020.
[6] J. Gomez, E. Kfoury, J. Crichigno, E. Bou-Harb, and G. Srivastava, "A performance evaluation of TCP BBRv2 alpha," in *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)*, 2020.
[7] Y. Song, G. Kim, I. Mahmud, W. Seo, and Y. Cho, "Understanding of BBRv2: evaluation and comparison with BBRv1 congestion control algorithm," *IEEE Access*, 2021.
[8] I. Baldin, A. Nikolich, J. Griffioen, I. Monga, K. Wang, T. Lehman, and P. Ruth, "FABRIC: A national-scale programmable experimental network infrastructure," *IEEE Internet Computing*, 2019.
[9] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 2010.
[10] B. Tierney, E. Dart, E. Kissel, and E. Adhikarla, "Exploring the BBRv2 congestion control algorithm for use on data transfer nodes," in *2021 IEEE Workshop on Innovating the Network for Data-Intensive Science (INDIS)*, 2021.
[11] S. Scherrer, M. Legner, A. Perrig, and S. Schmid, "Model-based insights on the performance, fairness, and stability of BBR," in *Proceedings of the 22nd ACM Internet Measurement Conference*, 2022.
[12] J. Crichigno, N. Ghani, J. Khoury, W. Shu, M. Wu, "Dynamic routing optimization in WDM networks," in *IEEE 2010 Global Telecommunications Conference (GLOBECOM)*, 2010.
[13] J. Crichigno, W. Shu, M. Wu, "Throughput optimization and traffic engineering in wdm networks considering multiple metrics," in *IEEE 2010 International Conference on Communications (ICC)*, 2010.
[14] N. Cardwell, Y. Cheng, S. Gunn, S. Yeganeh, and V. Jacobson, "BBR: congestion-based congestion control," *Communications of the ACM*, 2017.
[15] N. Cardwell, Y. Cheng, S. Yeganeh, P. Jha, Y. Seung, K. Yang, I. Swett, V. Vasiliev, B. Wu, and L. Hsiao, "BBRv2: A model-based congestion control performance optimization," in *Proc. IETF 106th Meeting*, 2019.
[16] E. Gavaletz and J. Kaur, "Decomposing RTT-unfairness in transport protocols," in *2010 17th IEEE Workshop on Local & Metropolitan Area Networks (LANMAN)*, 2010.
[17] Google, "TCP BBR v2 Alpha/Preview Release." [Online]. Available: https://github.com/google/bbr/tree/v2alpha, Accessed on 03-20-2023.
[18] J. Dugan, S. Elliott, B. Mah, J. Poskanzer, and K. Prabhu, "iPerf - The ultimate speed test tool for TCP, UDP, and SCTP." [Online]. Available: https://iperf.fr/, Accessed on 01-13-2023.
[19] S. Hemminger, "Network emulation with NetEm," 2005.
[20] N. Cardwell, Y. Cheng, S. Gunn, S. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control: Measuring bottleneck bandwidth and round-trip propagation time," *Queue*, 2016.
[21] M. Hock, R. Bless, and M. Zitterbart, "Experimental evaluation of BBR congestion control," in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, 2017.
[22] N. Rao, Q. Liu, S. Sen, J. Hanley, I. Foster, R. Kettimuthu, C. Wu, D. Yun, D. Towsley, and G. Vardoyan, "Experiments and analyses of data transfers over wide-area dedicated connections," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, 2017.
[23] 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*, 2018.
[24] FABRIC, "FABLib API." [Online]. Available: https://tinyurl.com/4dak7jc2, Accessed on 12-26-2022.
[25] P. Ruth, I. Baldin, K. Thareja, T. Lehman, X. Yang, and E. Kissel, "FABRIC network service model," in *2022 IFIP Networking Conference (IFIP Networking)*, 2022.
[26] Energy Sciences Network (ESNet), "Linux Tuning." [Online]. Available: https://tinyurl.com/y53742pz, Accessed on 01-13-2023.
[27] E. 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)*, 2019.
[28] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing router buffers," *ACM SIGCOMM Computer Communication Review*, 2004.
[29] J. Gettys, "Bufferbloat: Dark buffers in the internet," *IEEE Internet Computing*, 2011.