Measurements Comparing TCP Cubic and TCP BBR over a Satellite Network

Saahil Claypool
Worcester Polytechnic Institute
Worcester, MA, USA
smclaypool@wpi.edu

Jae Chung
Viasat
Marlborough, MA, USA
jaewon.chung@viasat.com

Mark Claypool
Worcester Polytechnic Institute
Worcester, MA, USA
claypool@cs.wpi.edu

***Abstract*—Satellite connections are critical for continuous network connectivity when disasters strike and for remote hosts that cannot use traditional network connections. While satellite Internet bitrates have increased, satellite latencies can still degrade TCP performance. Assessment of TCP over satellite networks is lacking, typically done only by simulation or emulation only, if at all. This paper presents experiments over a commercial satellite network comparing two popular TCP congestion control algorithms: Cubic (the default for most Internet servers) and BBR (recently deployed by Google servers). Analysis of the results shows similar steady-state bitrates for both BBR and Cubic, but with BBR having significantly higher bitrates (and, subsequently, higher round-trip times) than Cubic during start-up.**

***Index Terms*—congestion control, throughput, round-trip time**

I. Introduction

Satellite networks are an essential part of modern society, providing ubiquitous network connectivity even in times of disaster. There are over 2100 satellites in orbit, a 67% increase from 2014 to 2019 [1]. As few as three geosynchronous (GSO) satellites can cover the globe, interconnecting widely distributed ground-based networks and providing "last mile" connectivity to remote hosts. The idea of "always-on" connectivity is particularly useful for redundancy, especially in an emergency when traditional (i.e., wired) connections may not be available.

Geosynchronous orbit satellites have about 300 milliseconds of latency to bounce a signal up and down [2], a hurdle for TCP-based protocols that rely upon round-trip time communication to advance their data windows. TCP congestion control algorithms play a critical role in enhancing or restricting throughput in the presence of network loss and latency. TCP Cubic [3] is the default TCP congestion control algorithm in Linux and Microsoft Windows, but TCP BBR [4] has been widely deployed by Google on Linux servers and BBR is a congestion control option available in the QUIC transport protocol [5]. A better understanding of TCP congestion control algorithm performance over satellite networks is needed in order assess challenges and opportunities that satellite networks have to better support TCP (and QUIC) moving forward.

However, there are few published studies measuring network performance over actual satellite networks [6], with most studies either just using simulation [7] or emulation with satellite parameters [8], [9], [10], [11].

This paper presents results from experiments that measure the performance of TCP over a commercial satellite Internet network, comparing the default loss-based Cubic [3] with the bandwidth estimation-based BBR [4]. The network testbed and experiments are done over the Internet, but designed to be as comparable across runs as possible by interlacing sessions with each protocol serially to minimize temporal differences and by doing 80 bulk downloads for each protocol to provide for a large sample. In addition, a custom ping application provides several days worth of round-trip time and lost packet data to get a baseline on a "quiet" satellite network.

Analysis of the results shows the satellite link has consistent baseline round-trip times of about 600 milliseconds, but infrequently has round-trip times of several seconds. Loss events are similarly infrequent (less than 0.05%) and short-lived. Both Cubic and BBR have similar overall median throughputs, but BBR achieves the maximum link capacity more often than Cubic. Moreover, BBR has higher throughput during the start-up phase, meaning faster completion for short-lived downloads, such as Web pages. BBR and Cubic round-trip times are similar, indicating comparable amounts of queuing at the bottleneck link. However, both BBR and Cubic have periods of high retransmission rates owing to their oversaturation of the bottleneck queue.

The rest of this report is organized as follows: Section II describes research related to this work, Section III describes our testbed and experimental methodology, Section IV analyzes our experiment data, and Section V summarizes our conclusions and suggests possible future work.

II. Related Work

Ha et al. [3] develop TCP Cubic as an incremental improvement to earlier congestion control algorithms. TCP Cubic has been the default in Linux as of kernel 2.6.19 (in 2007), Windows 10.1709 Fall Creators update (in 2017), and Windows Server 2016 1709 update (in 2017).

Cardwell et al. [4] developed Bottleneck Bandwidth and Round-trip time (BBR) as an alternative to Cubic. BBR uses the maximum bandwidth and minimum round-trip time observed over a recent time window to build a model of the network and sets the congestion window size (to up to twice the bandwidth-delay product). BBR has been deployed by

Google servers since at least 2017 and is available for Linux TCP since Linux kernel 4.9 (end of 2016).

Cao et al. [12] analyze measurement results for BBR and Cubic over a range of different network conditions, and Ware et al. [13] model how BBR interacts with loss-based congestion control protocols (e.g., TCP Cubic). Both papers identify conditions which show performance benefits from BBR instead of Cubic.

Obata et al. [6] evaluate TCP performance over actual (not emulated, as is typical) satellite networks. Their experiments show throughputs around 26 Mb/s and round-trip times around 860 milliseconds.

Our work extends this work by providing detailed evaluation of Cubic and BBR in a satellite Internet network.

III. Methodology

A. Testbed

We setup a satellite link and configure our client and servers so as to allow for repeated tests. Our setup is design for comparative performance measurements by keeping all conditions the same across runs as much as possible, except for the change in TCP congestion control algorithm.

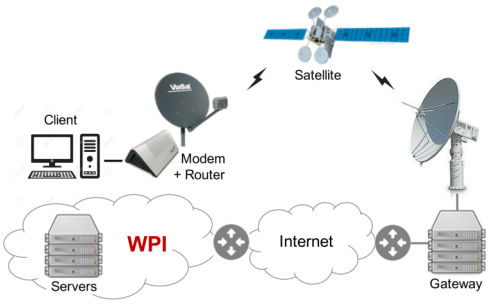

Fig. 1. Satellite measurement testbed.

Our testbed is depicted in Figure 1. The client is a Linux PC with an Intel i7-1065G7 CPU @ 1.30GHz and 32 GB RAM. There are two servers, one with Cubic and the other BBR. Both servers have an Intel Ken E312xx CPU @ 2.5 GHz and 32 GB RAM. The servers and client all run Ubuntu 18.04.4 LTS, Linux kernel version 4.15.0. The servers connect to our University LAN via a Gb/s Ethernet. The University campus network is connected to the Internet via several 10 Gb/s links, all throttled to 1 Gb/s.

The client connects to a Viasat satellite terminal (with a modem and router) via a Gb/s Ethernet connection.

The satellite terminal communicates through a Ka-band outdoor antenna (RF amplifier, up/down converter, reflector and feed) through the Viasat 2 satellite[1] to the larger Ka-band gateway antenna. The terminal supports adaptive coding and modulation using 16-APK, 8 PSK, and QPSK (forward) at 10 to 52 MSym/s and 8PSK, QPSK and BPSK (return) at 0.625 to 20 MSym/s. The satellite is capable of an aggregate throughput of up to 260 Gb/s.

[1]https://en.wikipedia.org/wiki/ViaSat-2

The terminal's connected Viasat service plan provides a peak data rate of 144 Mb/s.

The gateway does per-client queue management for traffic destined for the client, where the queue can grow up to 36 MBytes allowing a maximum queuing delay limit of about 2 seconds at the peak data rate. Queue lengths are controlled by Active Queue Management (AQM) that randomly drop incoming packets when the queue grows over half of the limit (i.e., 18 MBytes).

Wireshark captures all packet header data on each server and the client.

The performance enhancing proxy (PEP) that Viasat deploys by default is disabled for all experiments.

B. Baseline

For the network baseline, we run UDP Ping[2] consecutively for 1 week from a server to the client. UDP Ping sends one 20-byte UDP packet every 200 milliseconds (5 packets/s) from the server to the client and back, recording the round-trip time for each packet returned and the number of packets lost.

C. Downloads

We compare the performance of Cubic and BBR (version 1). The servers are configured for bulk-downloads via `iperf3`[3] (v3.3.1), one separate server for each congestion control algorithm. Cubic and BBR are used without further configuration.

For all hosts, the default TCP buffer settings are changed on both the server and client so that flows are not flow-controlled and instead are governed by TCP's congestion window. These include `tcp_mem`, `tcp_wmem` and `tcp_rmem` of 60 MBytes.

The client initiates a connection to one server via iperf, downloading 1 GByte of data, then proceeding to the next server. After cycling through both servers, the client pauses (1+ minutes) to allow the network to reset to baseline conditions. The process repeats a total of 80 times – thus, providing 80 network traces of a 1 GByte download for each protocol over the satellite link. Each cycle takes about 7 minutes, including the pause between cycles, for a total of about a day for all 80 cycles. We analyze results from a weekday in July 2020.

IV. Analysis

A. Network Baseline

We start by analyzing the network baseline loss and round-trip times, obtained on a "quiet" satellite link to our client – i.e., without any of our active bulk-downloads.

The vast majority (99%) of the round-trip times are between 560 and 625 milliseconds. However, the round-trip times have a heavy-tailed tendency, with 0.1% from 625 ms to 1500 ms and 0.001% from 1700 ms to 2200 ms. These high values show multi-second round-trip times can be observed on a satellite network even without any self-induced queuing. There are no visual time of day patterns to the round-trip times.

[2]http://perform.wpi.edu/downloads/#udp
[3]https://software.es.net/iperf/

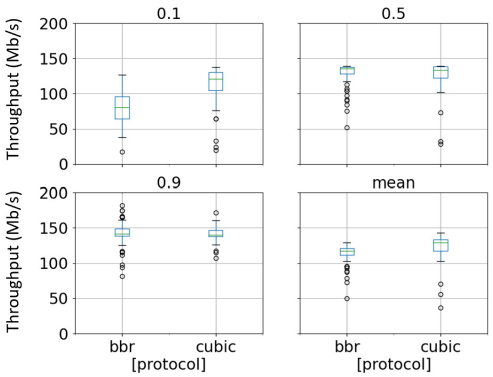

Fig. 2. Steady state throughputs.

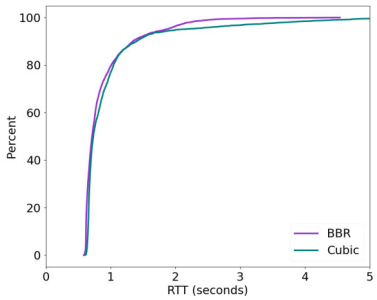

Fig. 3. Steady state round-trip times.

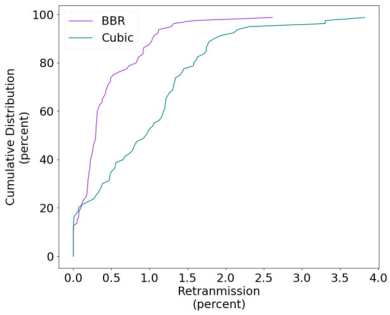

Fig. 4. Steady state retransmissions.

In the same time period, only 604 packets are lost, or about 0.05%. Most of these (77%) are single-packet losses, with 44 multi-packet loss events, the largest 11 packets (about 2.2 seconds). There is no apparent correlation between these losses and the round-trip times (i.e., the losses do not seem to occur during the highest round-trip times observed). These rates are considerably lower than the WINDS satellite loss of 0.7% [6].

B. Steady State

TCP's overall performance includes both start-up and congestion avoidance phases – the latter we call "steady state" in this paper. We analyze steady state behavior based on the last half (in terms of bytes) of each trace.

TABLE I
STEADY STATE THROUGHPUT.

Protocol	Mean (Mb/s)	Std Dev
BBR	112.9	12.2
Cubic	123.3	17.0

TABLE II
STEADY STATE ROUND-TRIP TIME.

Protocol	Mean (ms)	Std Dev
BBR	780	125.1
Cubic	821	206.4

Figure 2 depicts throughput comparisons for the steady state (last half) of all downloads for both protocols. The top left is the tenth percentile, the top right the 50% (or median), the bottom left the ninetieth percentile and the bottom right the mean. Each box depicts quartiles and median for the distribution. Points higher or lower than 1.4 $\times$ the inter-quartile range are outliers, depicted by the circles. The whiskers span from the minimum non-outlier to the maximum non-outlier. Table I shows the corresponding summary statistics.

From the graphs, BBR has a lower distribution of steady state throughput at the tenth percentile. This is attributed to the round-trip time probing phase by BBR, which, if there is no change to the minimum round-trip time, triggers every 10 seconds whereupon throughput is minimal for about 1 second. However, BBR has a higher distribution of throughput at the ninetieth percentile. BBR and Cubic have a similar median steady state throughputs, but BBR's varies less.

For the steady state throughputs, we do an independent, 2-tailed t test ($\alpha = 0.05$) and compute the effect size.[4] This test was found to be statistically significant ($t(158) = 4.4$, $p < .0001$) with a large effect size ($d = 0.7$).

Figure 3 shows the cumulative distributions of the round-trip times during steady state. The x-axis is the round-trip time in seconds computed from the TCP acknowledgments in the Wireshark traces, and the y-axis is the cumulative distribution. There is one trendline for each protocol. Table II shows the summary statistics.

BBR's round-trip time is somewhat lower than Cubic's and a bit steadier. Cubic, in particular, has a few cases with extremely high round-trip times. Across all flows, about 5% of the round-trip times are 2 seconds or higher.

Figure 4 shows the cumulative distributions of the retransmissions during steady state. From the figure, Cubic has the higher retransmission distribution.

C. Start-Up

We compare the start-up behavior for each protocol by analyzing the first 30 seconds of each trace, long enough to download about 50 MBytes on our satellite link. This is indicative of protocol performance for some short-lived flows, such as flows created during Web browsing.

The average Web page size for the top 1000 sites worldwide was around 2 MBytes as of 2018 [14], including HTML payloads, and all linked resources (e.g., CSS files and images). The distribution's 95th percentile was about 6 MBytes and the maximum was about 29 MBytes. Today's average total Web page size is probably about 5 MBytes [15], dominated by images and video.

Many long-lived TCP flows carry video content and these may be capped by the streaming video rate. However, assuming videos are downloaded completely, about 90% of YouTube videos are less than 30 MBytes [16].

The initial congestion window settings can vary from server to server on the Internet, ranging from 10 to 50 MSS for major CDN providers [17]. The default initial window in Linux since kernel version 2.6 in 2011 is 10. Our servers use the default initial congestion window of 10 for both BBR and Cubic.

Figure 5 depicts the time to download an object on the x-axis (in seconds) for an object size on the y-axis (in MBytes). The object size increment is 1 MByte. Each point is the

[4]Cohen's d quantifies mean differences in relation to the standard deviation. Small effect sizes are under 0.2, medium 0.2 to 0.5, and large 0.5 and above.

average time a protocol would require to download an object of the indicated size, shown with a 95% confidence interval.

From the figure, for the smallest objects (1 MByte), BBR downloads about 2 seconds faster than Cubic (7 versus 9 seconds). For an average Web page download (5 MBytes), BBR takes 10 seconds and Cubic takes 13 seconds. For 90% of all videos and the largest Web pages (30 MBytes), BBR takes 15 seconds and Cubic about 50% longer.

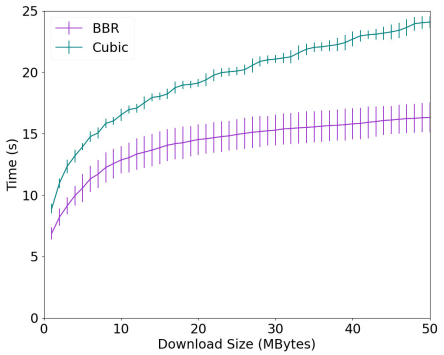

Fig. 5. Download time versus download object size.

Table III presents the summary statistics for the first 30 seconds of each flow for each protocol. During startup, Cubic has a low round-trip time, mostly because it takes a long time to ramp up its data rate, hence a low throughput. BBR has a higher round-trip time, with more packets in queue on average since it has more packets in flight.

TABLE III
START-UP SUMMARY STATISTICS.

	Tput (Mb/s)		RTT (ms)	
Protocol	Mean	Std Dev	Mean	Std Dev
BBR	23.1	1.8	917	42.9
Cubic	16.6	0.3	757	22.3

An independent, 2-tailed t test ($\alpha = 0.05$) for average start-up throughput was found to be statistically significant ($t(158) = 31.9$, $p < .0001$) with a large effect size ($d = 5$).

D. Limitations

Similar to other Internet studies, there are uncontrollable experimental dynamics that cannot be accounted for in our analysis, such as concurrent traffic on the wired (non-satellite) path and competition at the satellite gateway. To some extent, these concerns are mitigated by our back-to-back downloads that keep protocol measurements temporally close, and by doing 80 such downloads.

Time of day and day of week correlations have been observed on many Internet links [18]. While our measurements span a 24-hour period, thus ameliorating time of day effects, they are for a weekday only and may not accurately reflect a 24-hour period during a weekend day or holiday.

V. CONCLUSION

This paper presents results from experiments on a production satellite Internet network comparing TCP Cubic and TCP BBR. Results from 80 downloads for each protocol, interlaced so as to minimize temporal differences, are analyzed for steady state and start-up performance. Baseline satellite network results are obtained by long-term round-trip analysis in the absence of our bulk-download traffic.

Overall, the satellite link has consistent baseline round-trip times near the theoretical minimum (about 600 milliseconds) and very low (about a twentieth of a percent) loss rates. During steady state, Cubic and BBR have similar median throughputs, but Cubic has slightly higher mean throughput owing to BBR's bitrate reduction when probing for minimum round-trip times (probing for about one second, once per second). During start-up, BBR is about 50% faster than Cubic.

The results thus far are "in progress" since we are also comparing other congestion control algorithms, along with the effects of a PEP and multiple-flow scenarios.

REFERENCES

[1] S. I. Association, "Introduction to the Satellite Industry," Online presentation: https://tinyurl.com/y5m7z77e, 2020.

[2] Cisco, *Interface and Hardware Component Configuration Guide, Cisco IOS Release 15M&T*. Cisco Systems, Inc., 2015.

[3] S. Ha, I. Rhee, and L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," *ACM SIGOPS OS Review*, vol. 42, no. 5, 2008.

[4] N. Cardwell and Y. Cheng and C. S. Gunn and S. H. Yeganeh and Van Jacobson, "BBR: Congestion-based Congestion Control," *Communications of the ACM*, no. 2, pp. 58–66, Jan. 2017.

[5] N. Cardwell, Y. Cheng, S. H. Yeganeh, and V. Jacobson, "BBR Congestion Control," *IETF Draft*, Jul. 2017.

[6] H. Obata, K. Tamehiro, and K. Ishida, "Experimental Evaluation of TCP-STAR for Satellite Internet over WINDS," in *Proceedings of Autonomous Decentralized Systems*, Tokyo, Japan, Mar. 2011.

[7] C. Barakat, N. Chaher, W. Dabbous, and E. Altman, "Improving TCP/IP over Geostationary Satellite Links," in *Proceedings of GLOBECOM*, Rio de Janeireo, Brazil,, Dec. 1999.

[8] S. Utsumi, S. Muhammad, S. Zabir, Y. Usuki, S. Takeda, N. Shiratori, Y. Katod, and J. Kimb, "A New Analytical Model of TCP Hybla for Satellite IP Networks," *Network and Comp. Apps.*, vol. 124, Dec. 2018.

[9] Y. Wang, K. Zhao, W. Li, J. Fraire, Z. Sun, and Y. Fang, "Performance Evaluation of QUIC with BBR in Satellite Internet," in *Proceedings of IEEE Wireless for Space and Extreme Environments (WiSEE)*, Huntsville, AL, USA, Dec. 2018.

[10] V. Arun and H. Balakrishnan, "Copa: Practical Delay-Based Congestion Control for the Internet," in *Proceedings of the Applied Networking Research Workshop*, Montreal, QC, Canada, Jul. 2018.

[11] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "PCC: Re-architecting Congestion Control for Consistent High Performance," in *Proceedings of USENIX NSDI*, Oakland, CA, USA, 2015.

[12] Y. Cao, A. Jain, K. Sharma, A. Balasubramanian, and A. Gandhi, "When to Use and When not to Use BBR: An Empirical Analysis and Evaluation Study," in *Proceedings of Internet Measurement Conference (IMC)*, Amsterdam, NL, Oct. 2019.

[13] R. Ware, M. K. Mukerjee, S. Seshan, and J. Sherry, "Modeling BBR's Interactions with Loss-Based Congestion Control," in *Proceedings of Internet Measurement Conf. (IMC)*, Amsterdam, Netherlands, Oct. 2019.

[14] Data and Analysis, "Webpages Are Getting Larger Every Year, and Here's Why it Matters," Solar Winds Pingdom. Online at: https://tinyurl.com/y4pjrvhl, November 15 2018.

[15] T. Everts, "The Average Web Page is 3 MB. How Much Should We Care?" Speed Matters Blog. https://tinyurl.com/yas6k7zm, Aug. 9, 2017.

[16] X. Che, B. Ip, and L. Lin, "A Survey of Current YouTube Video Characteristics," *IEEE Multimedia*, vol. 22, no. 2, April - June 2015.

[17] CDN Planet, "Initcwnd Settings of Major CDN Providers," Online: https://tinyurl.com/ycjq63sq, feb 2017.

[18] P. Velan, J. Medkova, T. Jirsik, and P. Celeda, "Network Traffic Characterisation using Flow-based Statistics," in *IEEE/IFIP Network Ops. & Management Symp. (NOMS)*, Istanbul, Turkey, Apr. 2016.