RIPE
NCC
RIPE NETWORK
COORDINATION
CENTRE

THE SURPRISING IMPACT OF 1% PACKET LOSS

KEMAL ŠANJTA

PRINCIPAL INTERNET ANALYST

KEMALS@CISCO.COM

CISCO
ThousandEyes

RESEARCHED BUT NOT QUANTIFIED PROBLEM

- Intricacies of TCP are well researched
- Packet loss has negative effect on flows
- Not something that we quantify often
- Network engineers tend to look past “small” levels of packet loss (say 1 or 2%)

VARIOUS METHODS TCP USES TO HANDLE PACKET LOSS

- Duplicate ACKs
- Timeouts
- Explicit Congestion Notifications (ECN)
- Selective Acknowledgements (SACK)
- Congestion Avoidance Algorithms

CUBIC: THE DEFAULT CONGESTION AVOIDANCE ALGORITHM

- Given increased popularity of the Internet and growth of networks, network engineers realized that earlier congestion avoidance algorithms such as Tahoe, utilized available bandwidth slower than they should, especially in higher-bandwidth networks
- Default congestion avoidance algorithm on all major operating systems

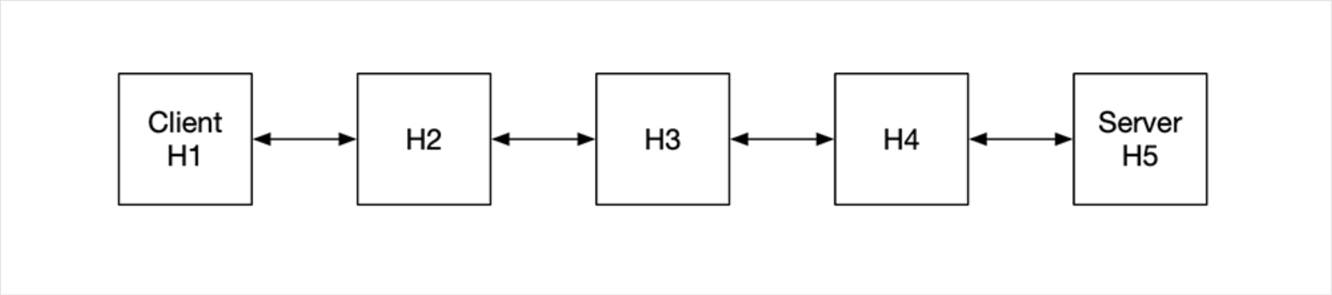
CUBIC: HOW IT WORKS?

- Congestion Window Adjustment
  - CUBIC employs a cubic function to adjust the congestion window size
  - The congestion window is increased aggressively during the slow start phase and cautiously during congestion avoidance. It reduces the congestion window sharply upon detecting packet loss, indicating network congestion
- Window Scaling
  - Adjusts the congestion window size based on the current network capacity and congestion level
- TCP Timestamps
  - CUBIC uses TCP timestamps for fine-grained measurement of round-trip time (RTT). Helps in estimating the available bandwidth and adjusting the congestion window accordingly
- Congestion Avoidance
  - Once the congestion window reaches a certain threshold, CUBIC switches to congestion avoidance mode. It increases the congestion window size gradually, probing for additional bandwidth without inducing congestion
- Packet Loss Reaction
  - CUBIC reacts to packet loss by reducing the congestion window size sharply
  - Implements an additive increase, multiplicative decrease (AIMD) approach to adjust the congestion window dynamically

TEST METHODOLOGY

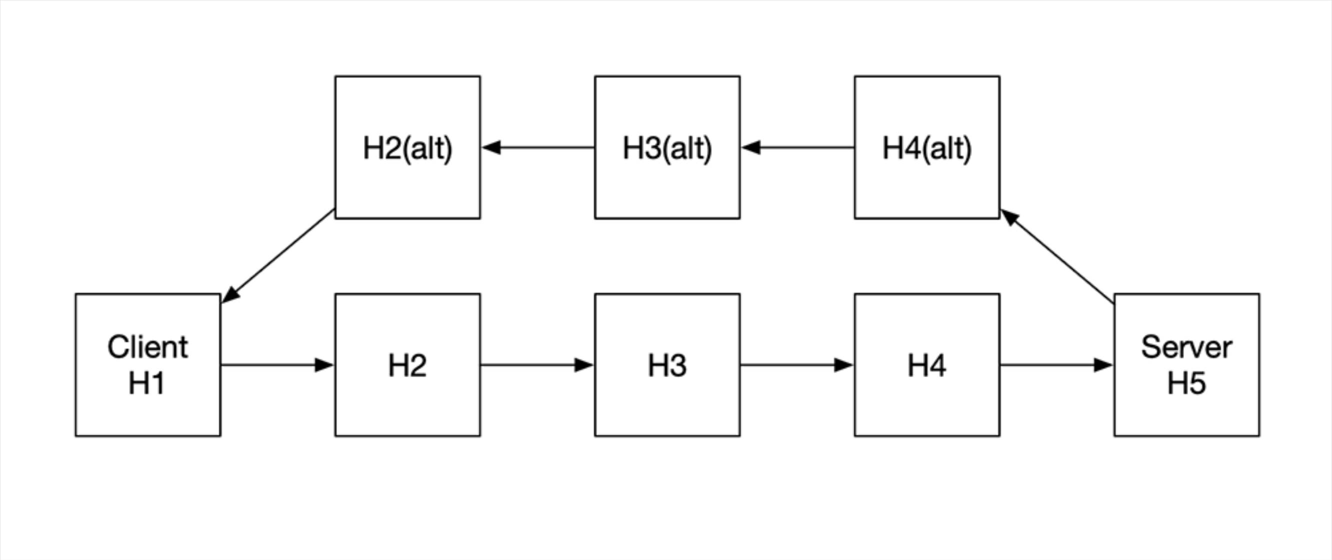
- Five Linux (Ubuntu 22.04) hosts configured to forward packets
- 1Gbps connectivity between devices
- Static routing
- Sub interfaces configured on hosts, required VLAN configuration on switch
- Measuring throughput using iperf3
- Unlike bandwidth, which represents the maximum capacity of the channel, throughput reflects the real-world performance and efficiency of the data transmission process

SYMMETRIC AND ASYMMETRIC NETWORK PATHS



Symmetric network (forward and reverse traffic path is the same)

Asymmetric network (reverse traffic is taking a different path when compared to the forwarding path)



ESTABLISHING A BASELINE (NO PACKET LOSS)

	Baseline (symmetric)
Mean	804.673506
STD	13.0217464
Min.	710
25%	799.99
50%	809.93
75%	810.046
Max.	830.419

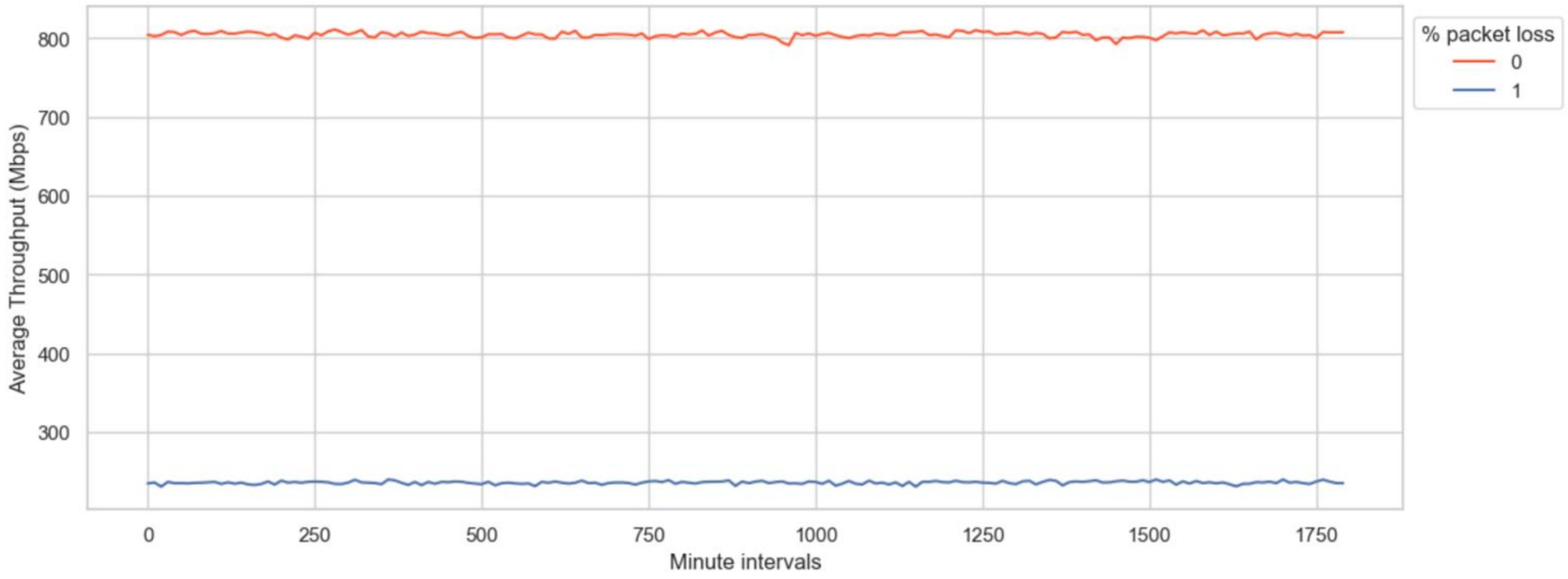
	Baseline (asymmetric)
Mean	864.139471
STD	14.647341
Min.	720.067
25%	859.973
50%	869.965
75%	870.3815
Max.	900.002

- 804.6 Mbps and 865.13 Mbps of Throughput for symmetric and asymmetric network, respectfully
- Asymmetric network traffic saw 7.3% increase in Throughput over symmetric network

INTRODUCING PACKET LOSS

- tc ("traffic control") utility
- tc has capabilities such as shaping, scheduling, policing, and dropping
- Enhancement called netem ("network emulation") that allows adding delay, packet loss, duplication, and other characteristics to packets outgoing from a specific network interface

THE CURIOUS CASE OF 1% PACKET LOSS



On average, 1% of packet loss causes 70.0%+ decrease in throughput!

- 804.6 Mbps of Throughput at baseline, 235.5 Mbps of Throughput at 1% loss in symmetric topology
- 864.13 Mbps of Throughput at baseline, 222.4 Mbps of Throughput at 1% loss in asymmetric topology

THE CURIOUS CASE OF 1% PACKET LOSS

	1% (symmetric)
Mean	235.513105
STD	13.5692798
Min.	93.967
25%	229.667
50%	236.635
75%	243.596
Max.	281.886

	1% (asymmetric)
Mean	222.493196
STD	13.7883065
Min.	51.21
25%	214.788
50%	222.729
75%	230.675
Max.	280.877

1% of packet loss caused a 70.7% decrease in throughput in symmetric network topology, while in asymmetric topology it resulted in 74.2% decrease in throughput!

OVERALL RESULTS

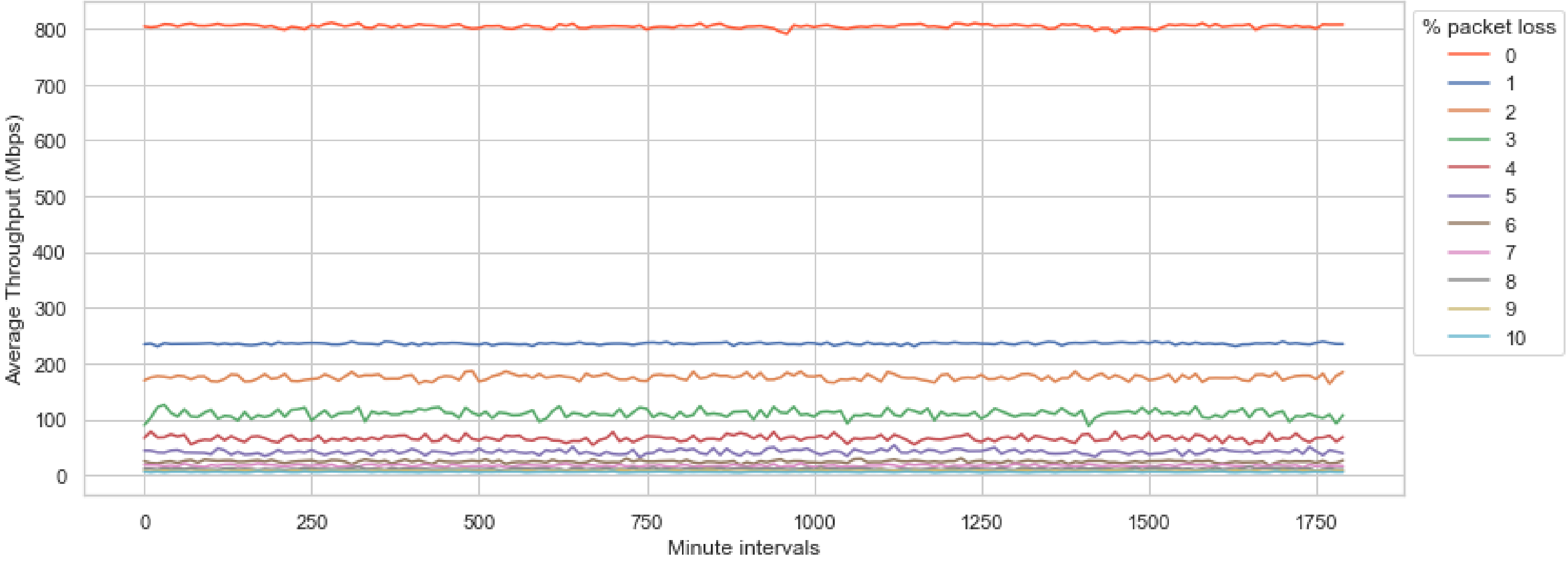
Symmetric network

	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Mean	235.51	175.19	109.76	65.68	41.37	23.95	16.75	11	7.52	5.29
STD	13.57	37.48	46.68	36.09	25.48	17.31	12.16	8.4	5.97	4.33
Min.	93.97	11.93	0	0	0	0	0	0	0	0
0.25	229.67	158.09	74.56	37.77	21.38	9.94	6.96	4.97	2.98	1.99
0.5	236.64	190.91	111.86	61.67	37.77	19.89	13.92	8.95	5.97	3.98
0.75	243.6	199.86	150.14	89.53	57.18	33.81	23.37	15.41	9.95	6.96
Max.	281.89	223.72	201.33	175.49	149.62	119.3	87.5	68.59	46.76	37.78

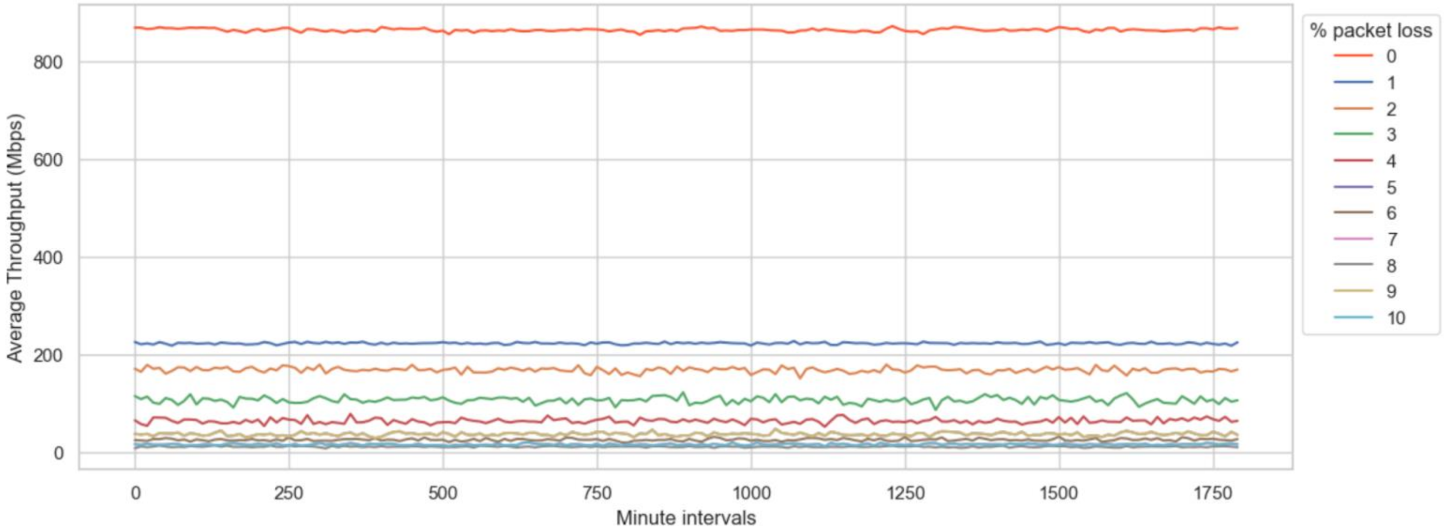
Asymmetric network

	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Mean	222.49	168.03	106.43	63.57	36.59	24.99	15.52	10.82	36.59	15.52
STD	13.79	34.91	44.62	34.81	24.44	16.93	11.58	8.26	24.44	11.58
Min.	51.21	5.97	0	0	0	0	0	0	0	0
25%	214.79	151.14	72.57	35.8	16.9	11.93	5.97	4.97	16.9	5.97
50%	222.73	182.45	108.35	59.66	31.84	21.87	11.94	8.95	31.84	11.94
75%	230.68	191.89	144.67	87	51.7	34.79	21.87	14.92	51.7	21.87
Max.	280.88	212.79	188.91	163.07	148.64	118.81	82.03	63.64	148.64	82.03

OVERALL RESULTS VISUALISED



Throughput achieved in symmetric network



Throughput achieved in asymmetric network

BBR: THE FUTURE OF CONGESTION AVOIDANCE?

- BBR stands for Bottleneck Bandwidth and Round-Trip Time
- It is a congestion control algorithm developed by Google
- Designed to optimize network utilization and throughput by continuously probing for the available bandwidth and adjusting sending rate accordingly

BBR: HOW IT WORKS?

- Bandwidth estimation
  - BBR estimates the available bandwidth by measuring the delivery rate of packets
  - Uses concept of pacing to ensure a steady flow of packets without causing undue congestion
- Round-Trip Time (RTT) Estimation
  - Maintains an estimate of the minimum RTT of the connection
  - RTT variations are used to adjust the pacing rate, ensuring smooth transmission and reduced latency
- Bottleneck Detection
  - Identifies the bottleneck link in the network path through various techniques like probing for increased delivery rates and utilizing RTT feedback
- Congestion Window Management
  - Adjusts the sending rate by maintaining two parameters: pacing gain and probing gain
- Low Latency Operation
  - Aims to keep the queue size low, which helps in reducing latency

KEY DIFFERENCES BETWEEN CUBIC AND BBR

- Congestion Window Adjustment
  - **CUBIC**: Adjusts congestion window based on cubic function, reacting strongly to loss
  - **BBR**: Dynamically adjusts sending rate based on bandwidth and RTT estimations, avoiding unnecessary loss
- Bandwidth Estimation
  - **CUBIC**: Relies on packet loss as an indicator of congestion
  - **BBR**: Actively probes for available bandwidth and adjusts sending rate, minimizing latency
- Latency Optimization
  - **CUBIC**: Prioritizes throughput over latency, potentially leading to increased latency under heavy congestion
  - **BBR**: Maintains low latency by continuously monitoring network conditions and adjusting congestion control parameters accordingly
- Implementation
  - **CUBIC**: Widely adopted in many operating systems and network devices
  - **BBR**: Developed by Google for its data centers, gaining adoption in various platforms and protocols.

ENABLING BBR

```
cat /proc/sys/net/ipv4/tcp_congestion_control
cubic
```

Verify currently configured algorithm

```
echo "net.core.default_qdisc=fq" >> /etc/sysctl.conf
echo "net.ipv4.tcp_congestion_control=bbr" >> /etc/sysctl.conf
sysctl -p
```

Enable BBR

```
cat /proc/sys/net/ipv4/tcp_congestion_control
bbr
```

Verify that BBR is configured

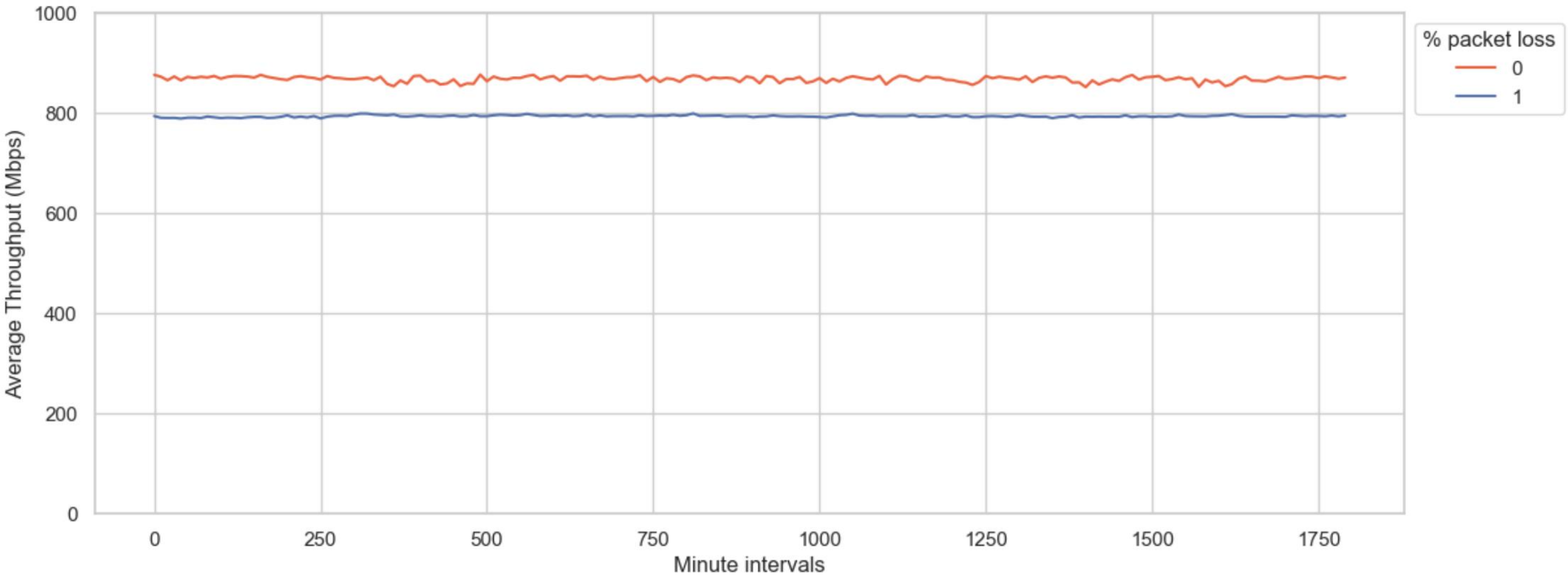
ESTABLISHING A BASELINE WITH BBR (NO PACKET LOSS)

	Baseline (symmetric)
Mean	868.50
STD	49.36
Min.	679.99
25%	860.15
50%	889.99
75%	890
Max.	900.31

	Baseline (asymmetric)
Mean	827.20
STD	46.06
Min.	639.99
25%	839.92
50%	840
75%	849.99
Max.	860.26

- 868.5 Mbps and 827.20 Mbps of Throughput for symmetric and asymmetric network, respectfully
- Asymmetric network traffic saw 4.7% decrease in Throughput over symmetric network

MEASURING IMPACT OF 1% PACKET LOSS WHILE USING BBR



On average, 1% of packet loss caused 8.5% decrease in throughput while using BBR, stark difference to 70.7% decrease using CUBIC!

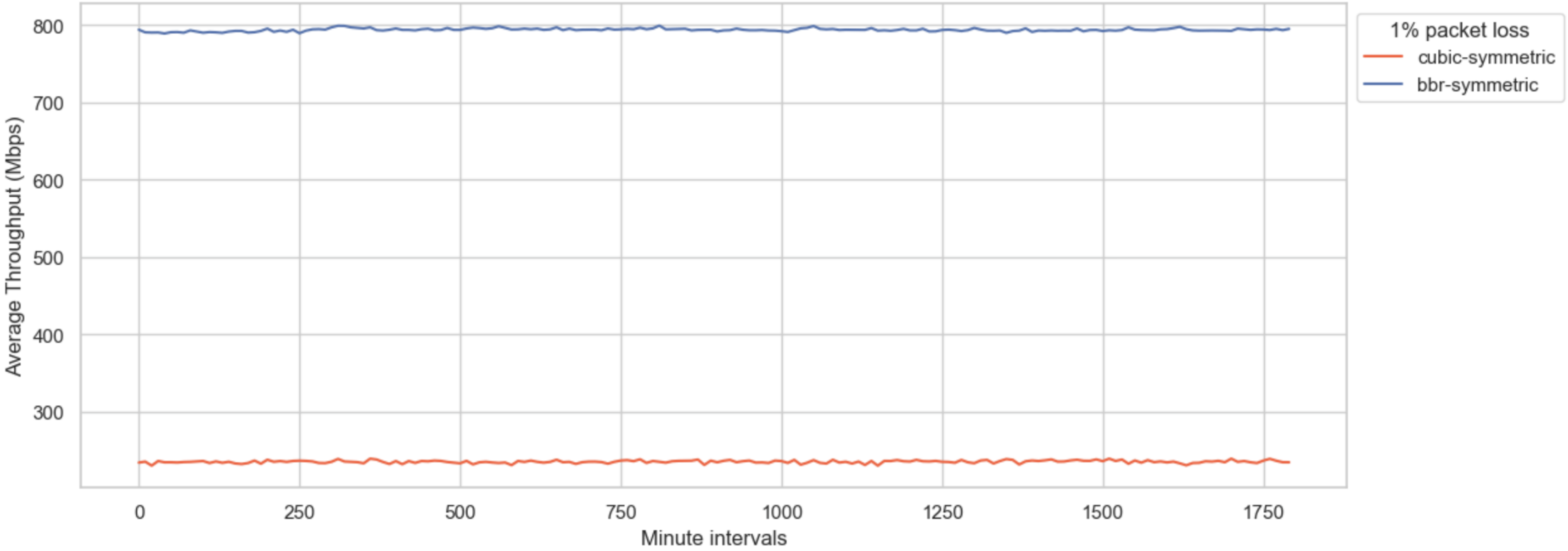
MEASURING IMPACT OF 1% PACKET LOSS WHILE USING BBR

	1% (symmetric)
Mean	794.06
STD	44.08
Min.	489.99
25%	800.33
50%	809.99
75%	810.01
Max.	830.08

	1% (asymmetric)
Mean	763.42
STD	44.28
Min.	519.96
25%	760
50%	779.99
75%	789.98
Max.	810.41

- 1% packet loss, in symmetric network topology using BBR, caused 8.5% throughput decrease compared to 70.7% throughput decrease in the same topology while using CUBIC
- In asymmetric network topology using BBR, we saw 7.7% throughput decrease compared to 74.2% decrease in throughput while using CUBIC

COMPARISON BETWEEN CUBIC AND BBR AT 1% LOSS



OVERALL RESULTS WITH BBR

Symmetric network

	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Mean	794.06	791.65	768.94	775.34	773.7	787.71	784.07	644.04	761.61	751.89
STD	44.08	44.58	47.55	50.11	56.29	61.42	64.99	268.31	76.86	77.96
Min	490	370	140	280.05	209.86	0	130	0	0	0
25%	800.34	799.99	779.99	780.27	788.9	800	799.99	750.01	780	770
50%	810	809.93	780.02	790	790	800.92	800	769.99	780.03	770.8
75%	810.01	810	790	790.2	790.25	810	810	770.02	790	780
Max	830.09	830.76	810.53	831.33	820.09	831.26	830.07	800.09	810.07	800.2

Asymmetric network

	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Mean	763.42	822.11	795.6	812.53	792.47	793.79	750.63	749.33	760.8	751.64
STD	44.28	46.83	48.91	53.64	57.29	62.64	63.99	68.44	73.83	81.68
Min	519.96	500	270	249.83	160	39.98	0	0	0	0
25%	760.01	830	800.02	820.01	800.33	809.6	760.01	760	779.98	770
50%	780	839.99	810	830	810	810	770	770	780.01	779.77
75%	789.99	840.01	819.98	830.07	811.09	819.98	770.05	770.03	790	780.01
Max	810.42	860.04	840.08	850.17	840	840	820	810.09	810.14	800.07

BBR PRODUCTION TESTING

- Single POP (Tokyo) testing at Dropbox
  - Performance comparison between BBRv1 and BBRv2
  - Performance comparison with CUBIC and Reno
  - Results indicate production readiness
- Subset of Spotify users
  - Results indicate production readiness
- Google
  - They built it for their use case, kind of expected
- Reports of Netflix working with BBR on FreeBSD
- Cisco Catalyst SD-WAN enables it between SD-WAN endpoints when “tcp-optimization” feature is selected

CONCLUSION

- Even the smallest amount of packet loss has extremely negative consequences on throughput
- Outlines importance of monitoring and addressing even minor levels of packet loss
- CUBIC is, still, default congestion avoidance algorithm
- Packet loss outcomes significantly differ based on congestion avoidance algorithm used
- BBR shows significantly better results at any packet loss %