ANALYSIS AND EVALUATION OF TCP CONGESTION ALGORITHMS

Lawrence Brakmo

Facebook

brakmo@fb.com

INTRODUCTION

- Congestion control is a very hard problem
- People have been working at it for many decades
- Algorithm needs to utilize available bandwidth
  - Fairly
  - When many unrelated flows are competing
- Consider start up time. We do not know available bw
  - What if we did? What could we do with that information?
  - Jump to that rate? NO – there may be other flows starting up and getting the same information

FOCUS

- I will focus on the following congestion algorithms
- *Reno* – the grandfather of all, although it has been improved
- *Cubic* – the default in Linux. Better than Reno for WAN traffic. Has hystart.
- *DCTCP* – Uses ECN markings to achieve congestion avoidance. Much better than TCP's default ECN behavior. Only good for Data Centers
- *BBR* – The new player in town. Still lots of questions about it.
- *NV* – A follow up to Vegas (my babies). Only tuned for Data Centers using TCP-BPF to set baseRTT to 80us
- *TCP-BPF* – Cubic using TCP-BPF to clamp cwnd. Only for D. C.

FOCUS (2)

- Then in the last couple of days I added the following for WAN tests
- *BIC*
- *Yeah*
- *HighSpeed*
- *H-TCP*
- *Westwood*

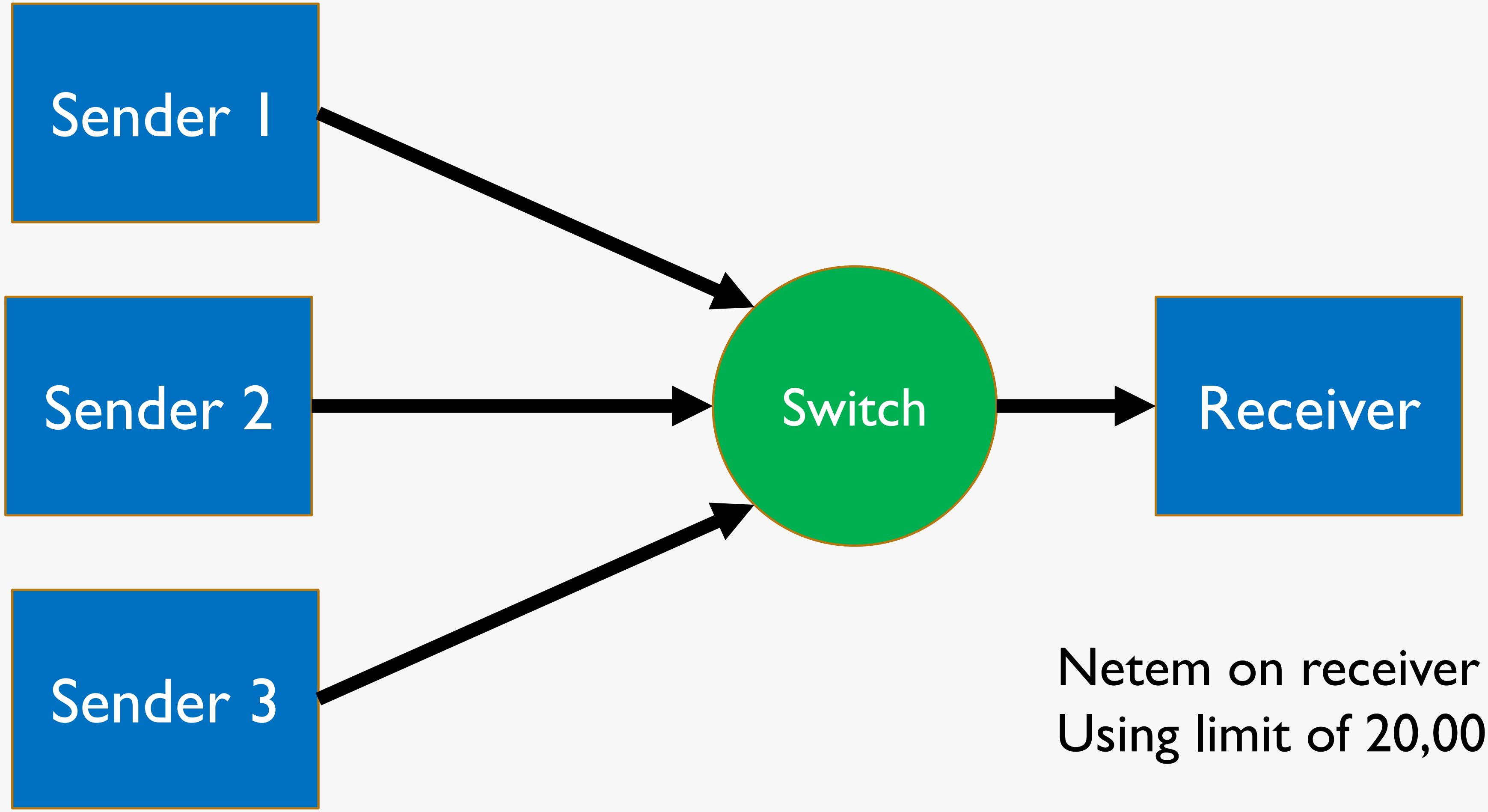
CONGESTION VS. AVOIDANCE

- *Reno* and *Cubic* do not avoid congestion. On the contrary, they periodically create congestion and losses. It is the only way they can know they have reached full bandwidth use.

- *DCTCP, BBR** and *NV* do congestion avoidance. They detect, or try to, congestion before losses occur. And in many cases they can keep buffers quite small improving latency

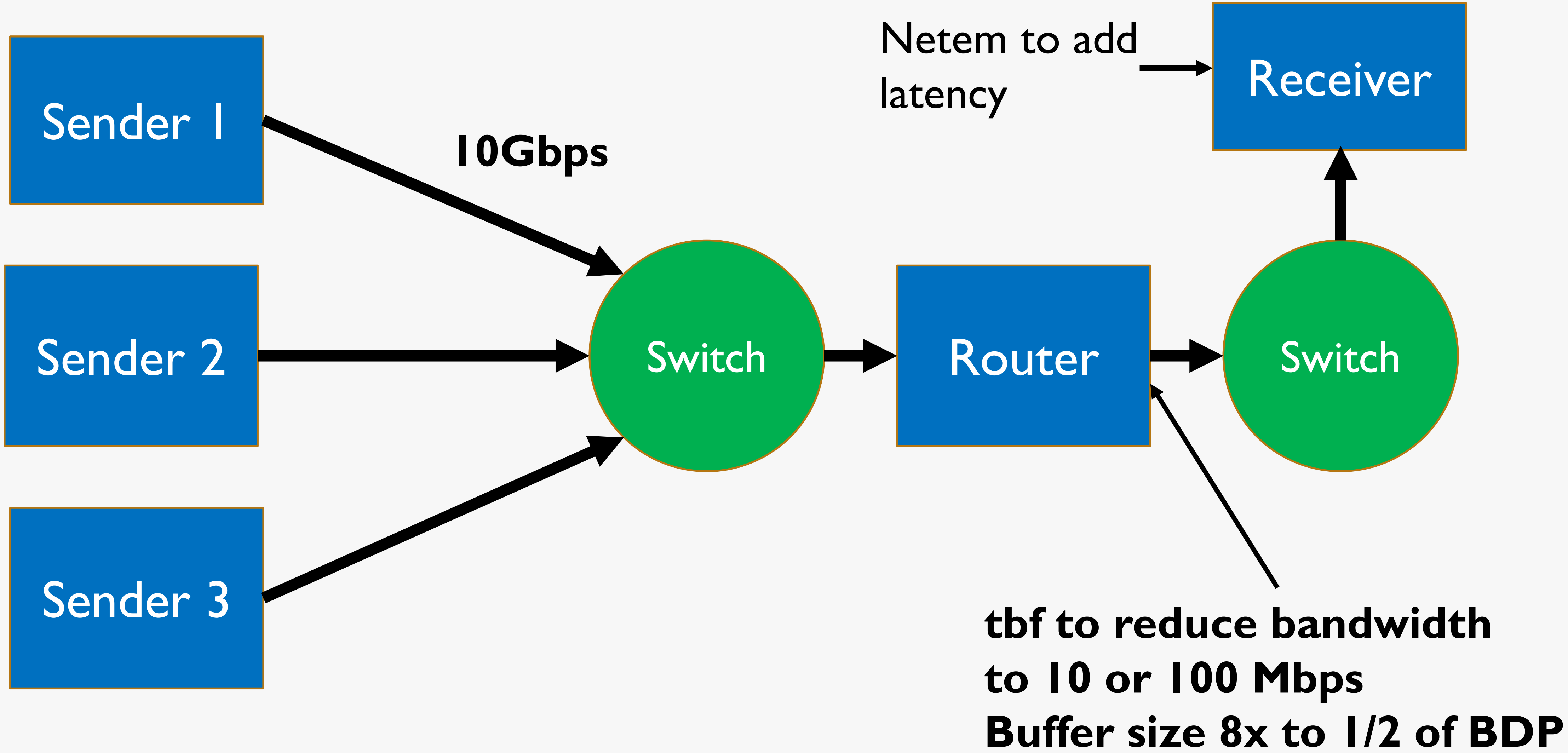
- No losses => better high percentile latencies.

EXPERIMENTAL SETUP FOR 10G TESTS



Netem on receiver when adding latency
Using limit of 20,000

EXPERIMENTAL SETUP FOR 10 AND 100 MPBS TESTS



Netem to add latency

tbf to reduce bandwidth to 10 or 100 Mbps
Buffer size 8x to 1/2 of BDP

EXPERIMENTAL SETUP (2)

- Scenarios
  - LAN with 20us RTT, 10 Gbps - servers in same rack.
  - Fast WAN with 10ms RTT, 10 Gbps
  - WAN with 40ms RTT, 10 and 100 Mbps
- Tests
  - *Fairness & Stability* - consists of 2 or 3 stream flow tests (each from a different server) to one receiver
  - *Size Fairness* – consists of a combination of streaming, 1MB and 10KB RPCs (8MB and 1MB for 10G-10ms scenario)

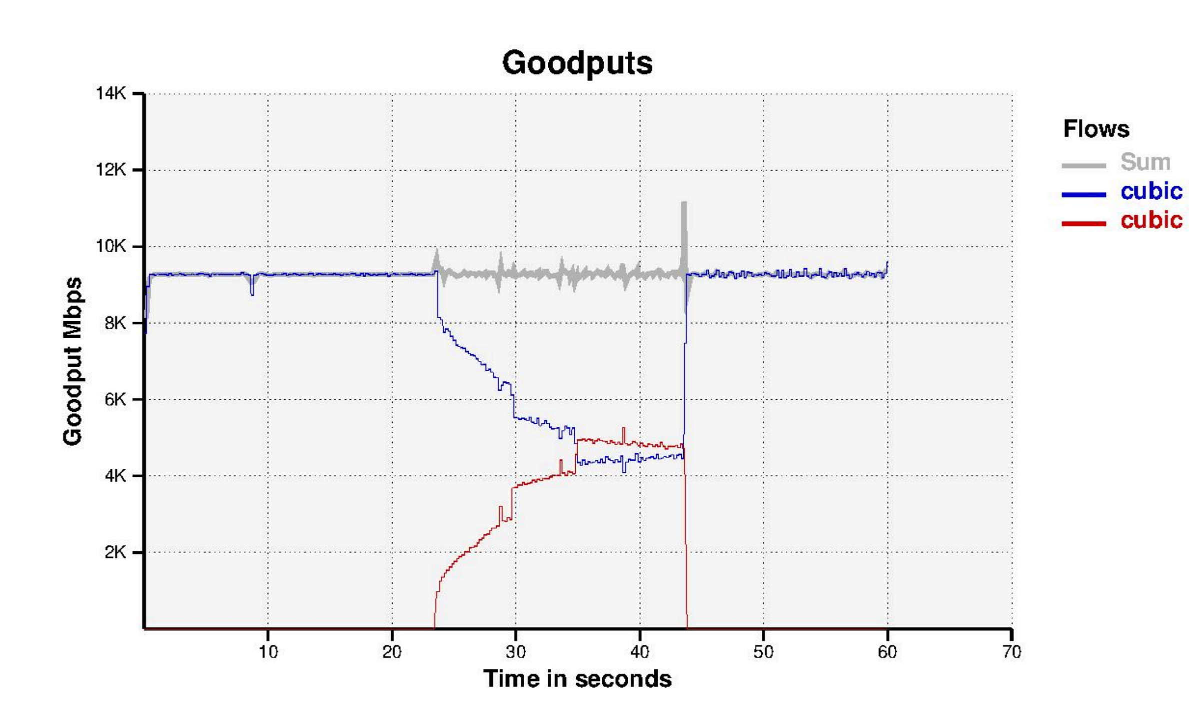
EXPERIMENTAL SETUP

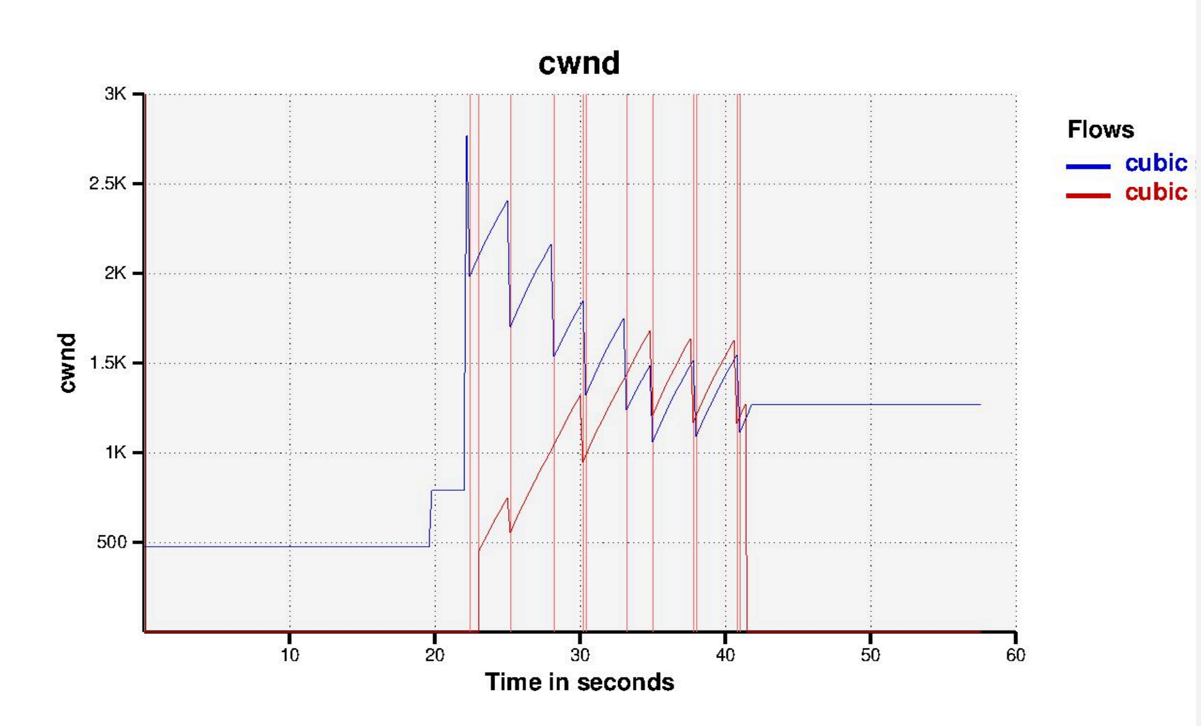
- Netesto is used to run the experiments, collect the data and create graphs and tables
  - Graphs of goodput, cwnd, RTTs, minRTTs, retransmissions
  - Tables with all the details (Goodputs, RTTs, cwnd, latencies, retransmissions, etc.
- Used Linux kernel 4.14.0-rc5
- Used mq and fq_codel queuing disciplines.
- For DCTCP and NV switch has 2 queues, one for DCTCP with ECN enable or NV, one for everything else

RESULTS

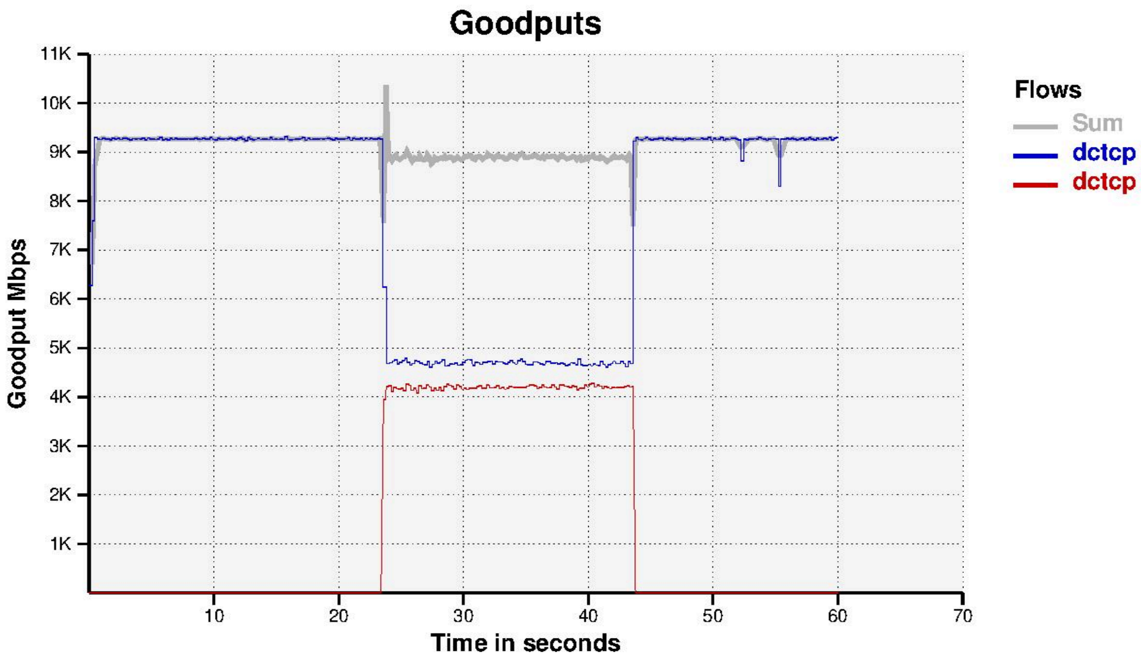
10G LAN 2 FLOWS

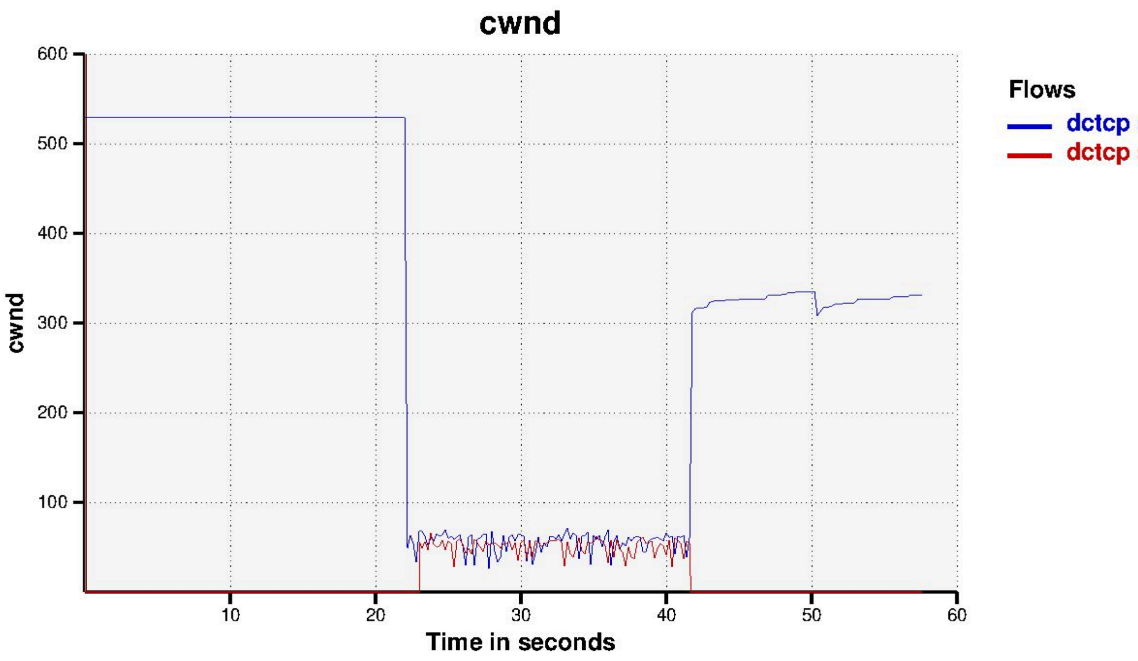
CUBIC


Goodputs
14K
12K
10K
8K
6K
4K
2K
Goodput Mbps
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
cubic
cubic

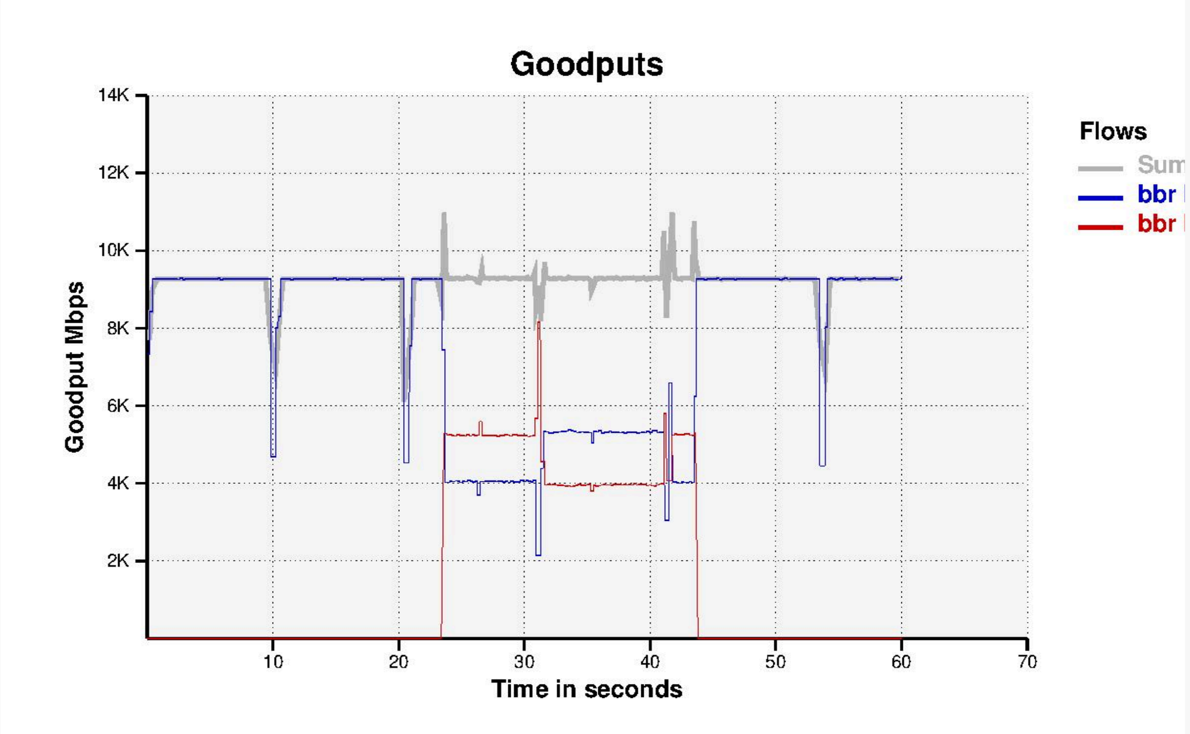

cwnd
3K
2.5K
2K
1.5K
1K
500
cwnd
10
20
30
40
50
60
Time in seconds
Flows
cubic
cubic

DCTCP

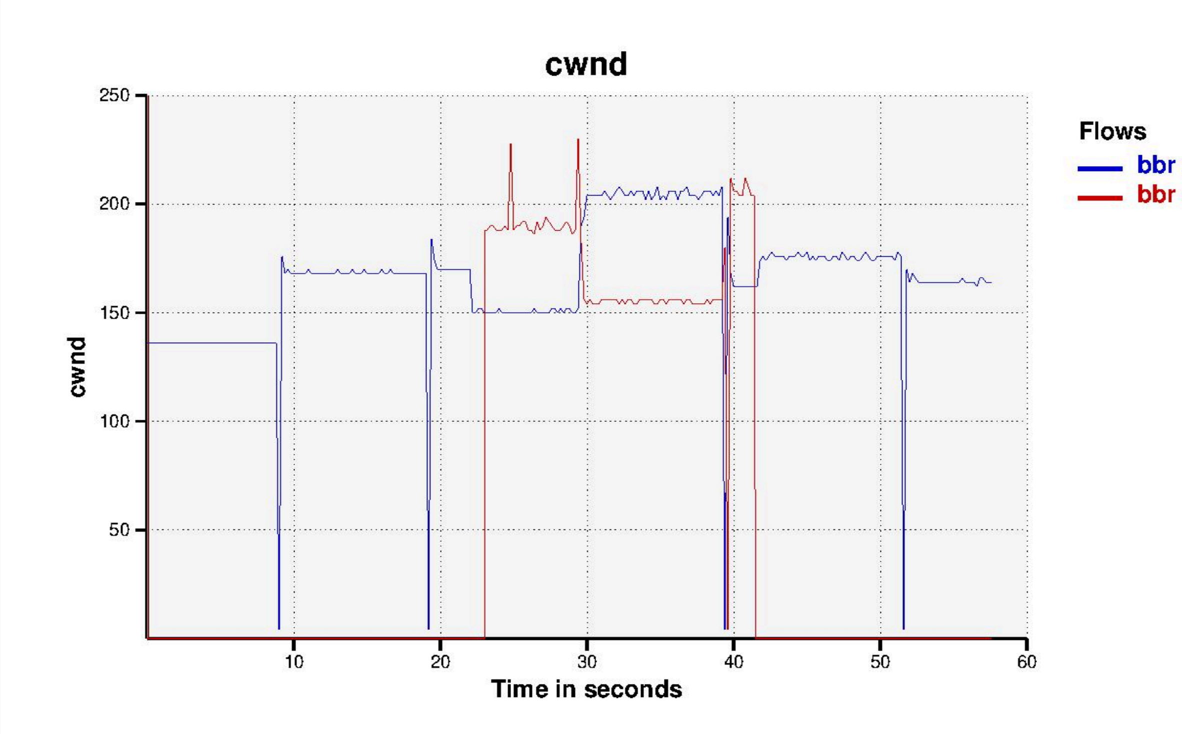

Goodputs
Goodput Mbps
11K
10K
9K
8K
7K
6K
5K
4K
3K
2K
1K
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
dctcp
dctcp


cwnd
cwnd
600
500
400
300
200
100
10
20
30
40
50
60
Time in seconds
Flows
dctcp
dctcp

BBR

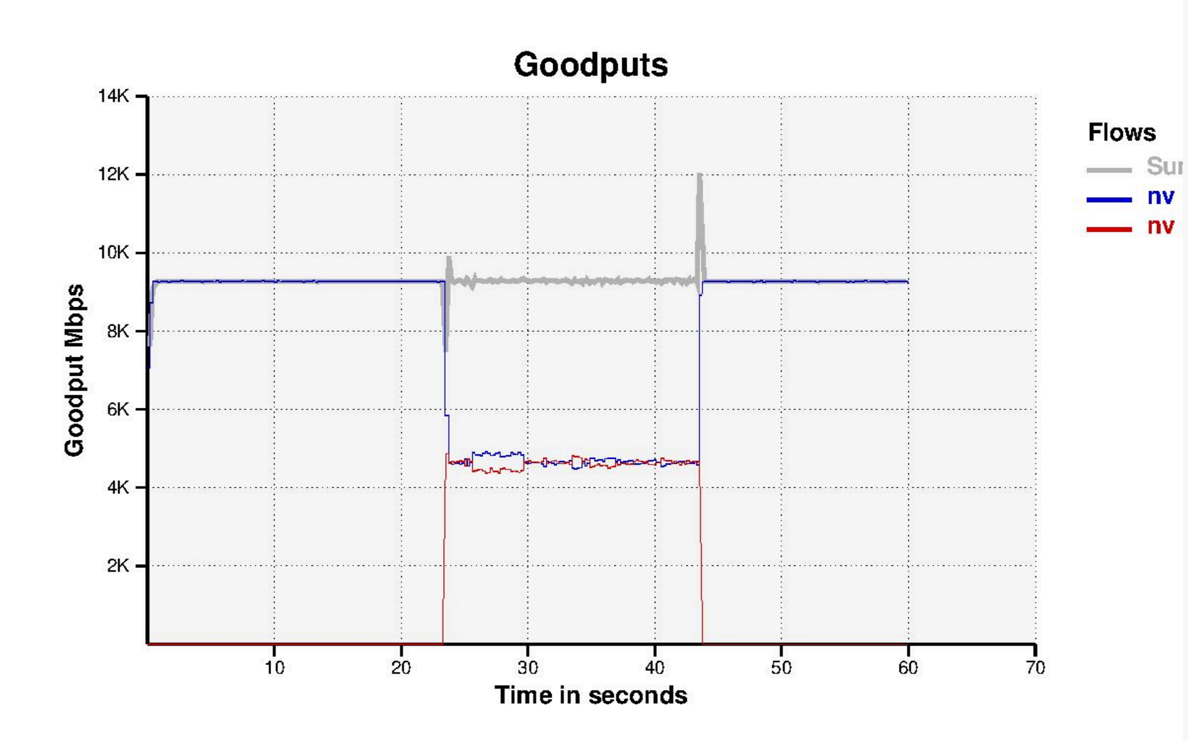


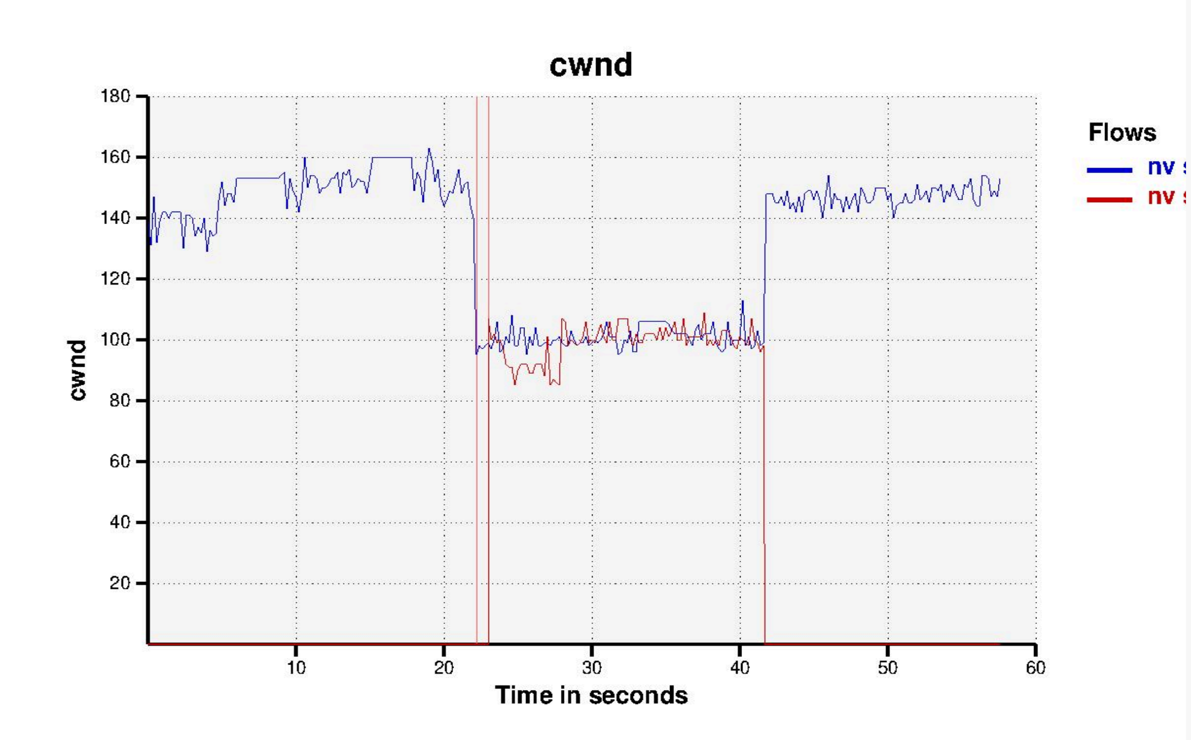
Goodputs
Goodput Mbps
14K
12K
10K
8K
6K
4K
2K
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
bbr
bbr



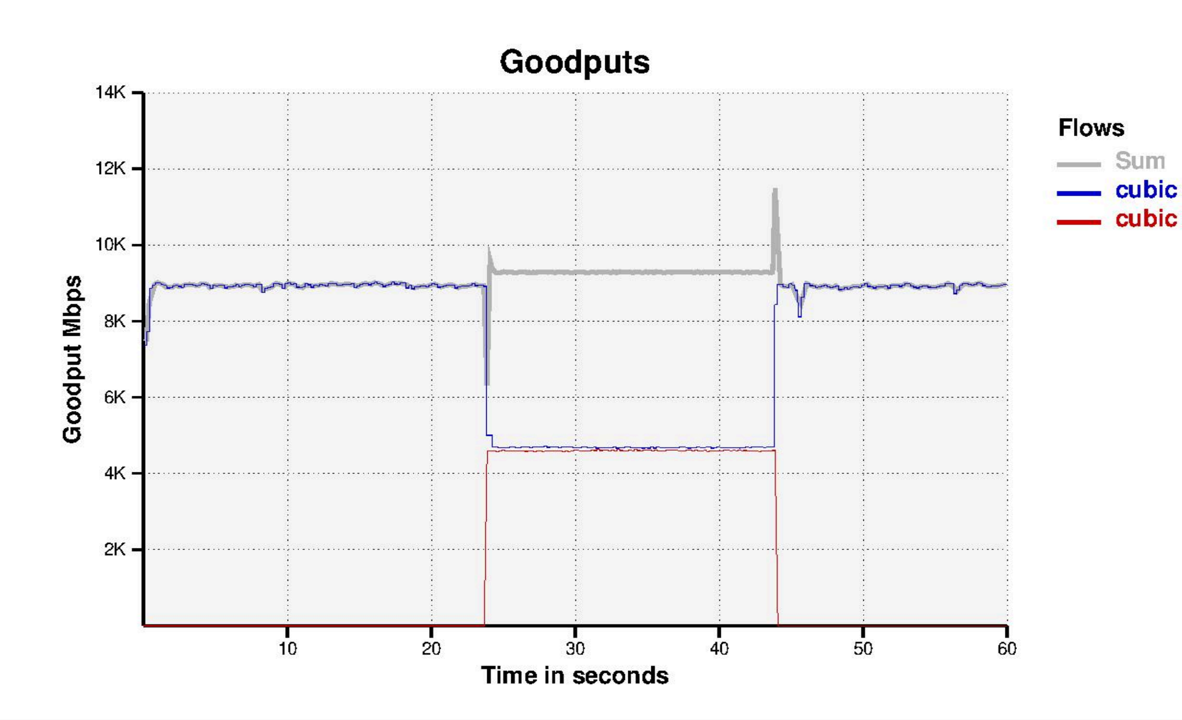
cwnd
cwnd
250
200
150
100
50
10
20
30
40
50
60
Time in seconds
Flows
bbr
bbr

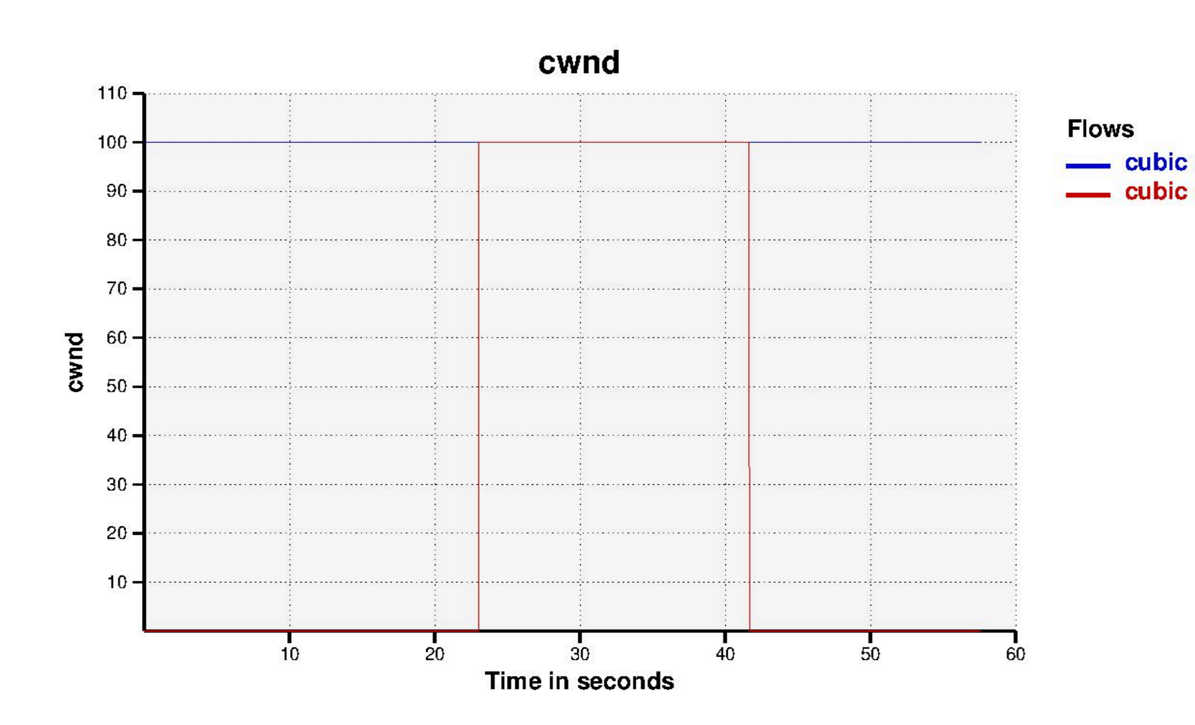
NV W/BASERTT OF 80US





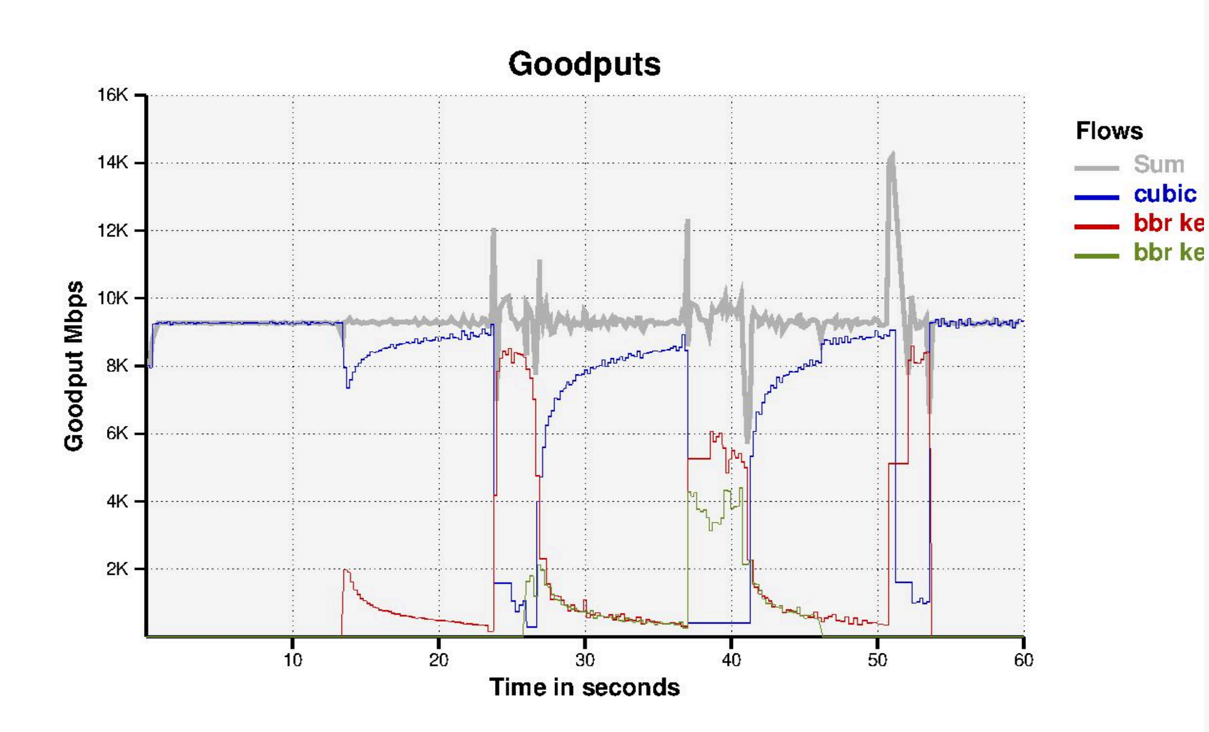
CUBIC W/TCP-BPF

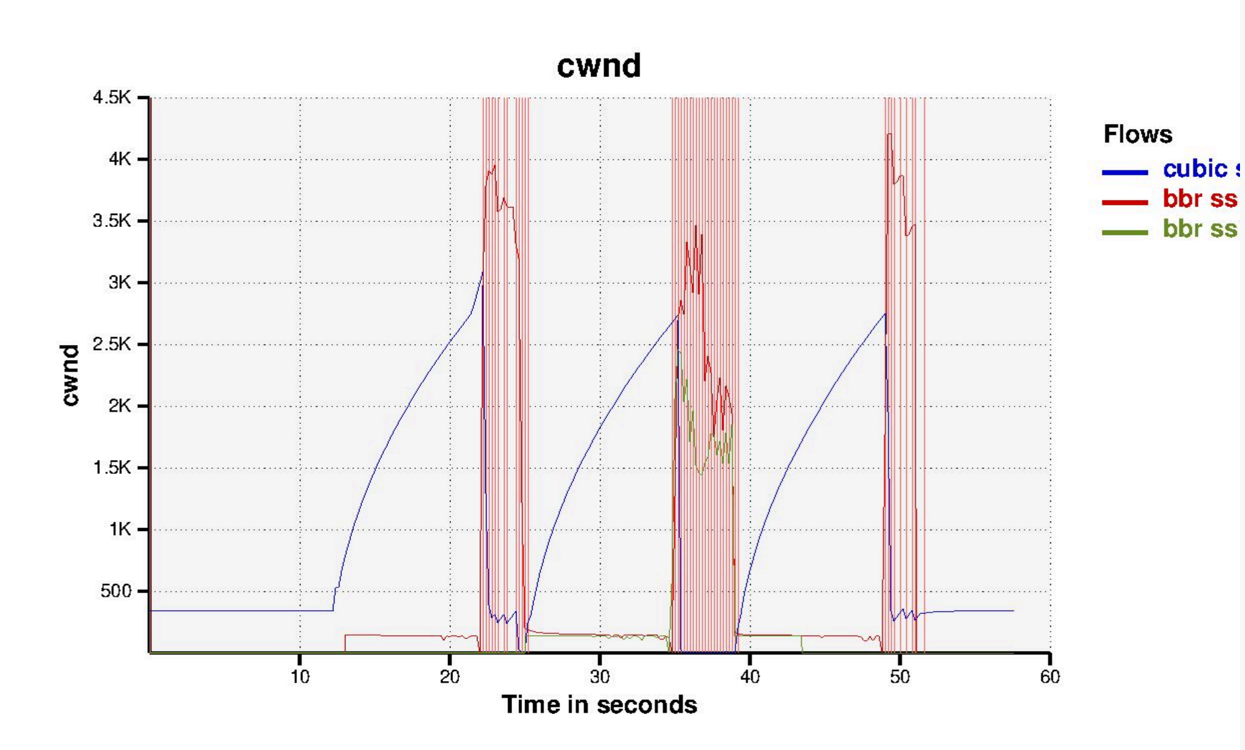




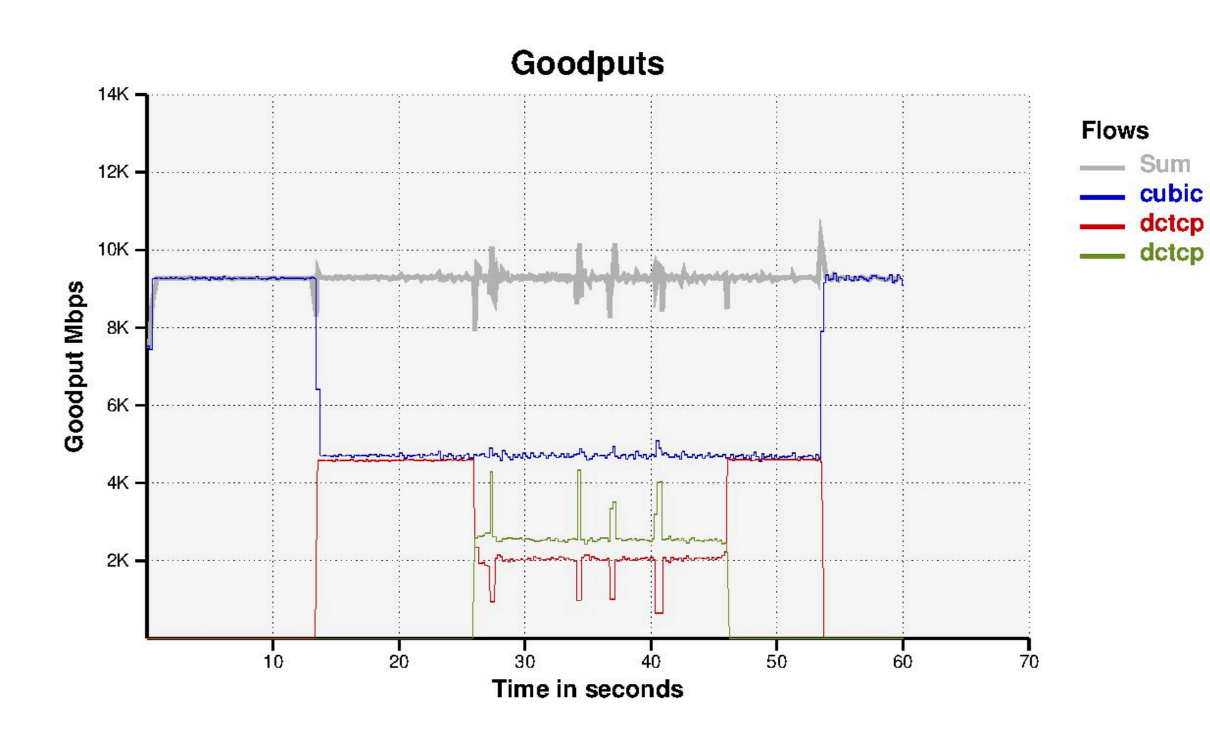
3-FLOWS, 1-CUBIC VS. 2..

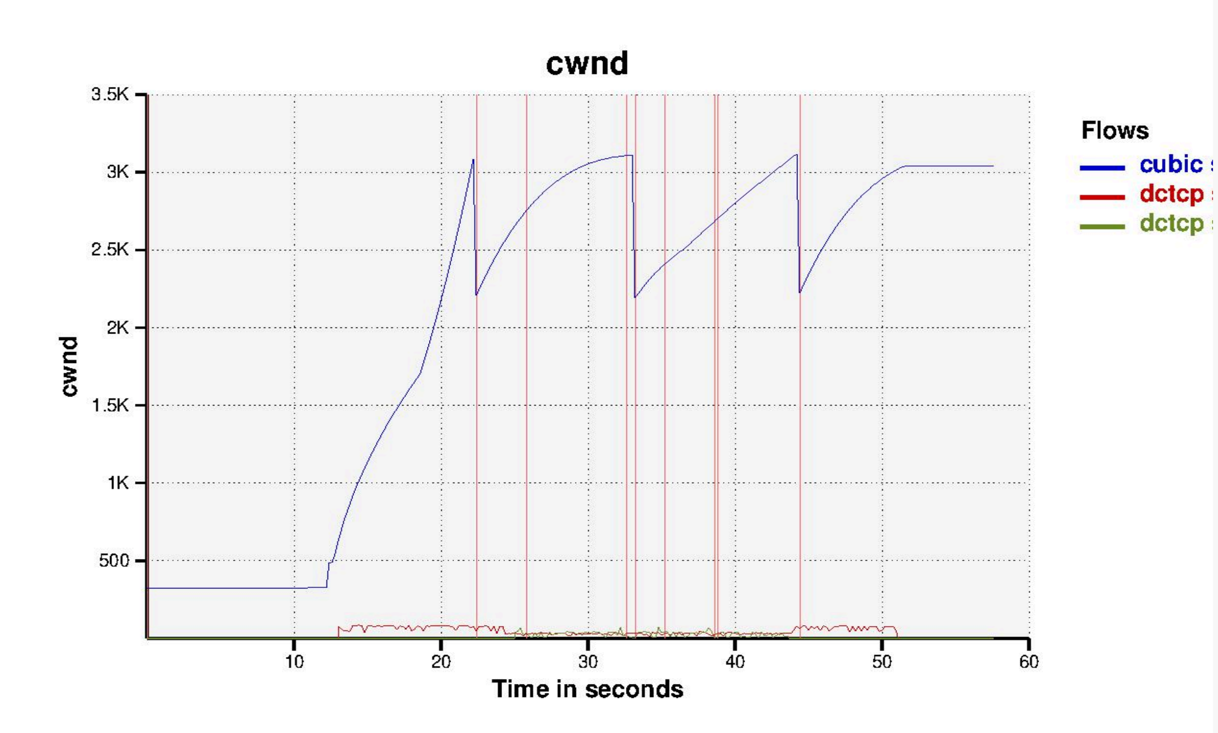
1-CUBIC VS. 2-BBR



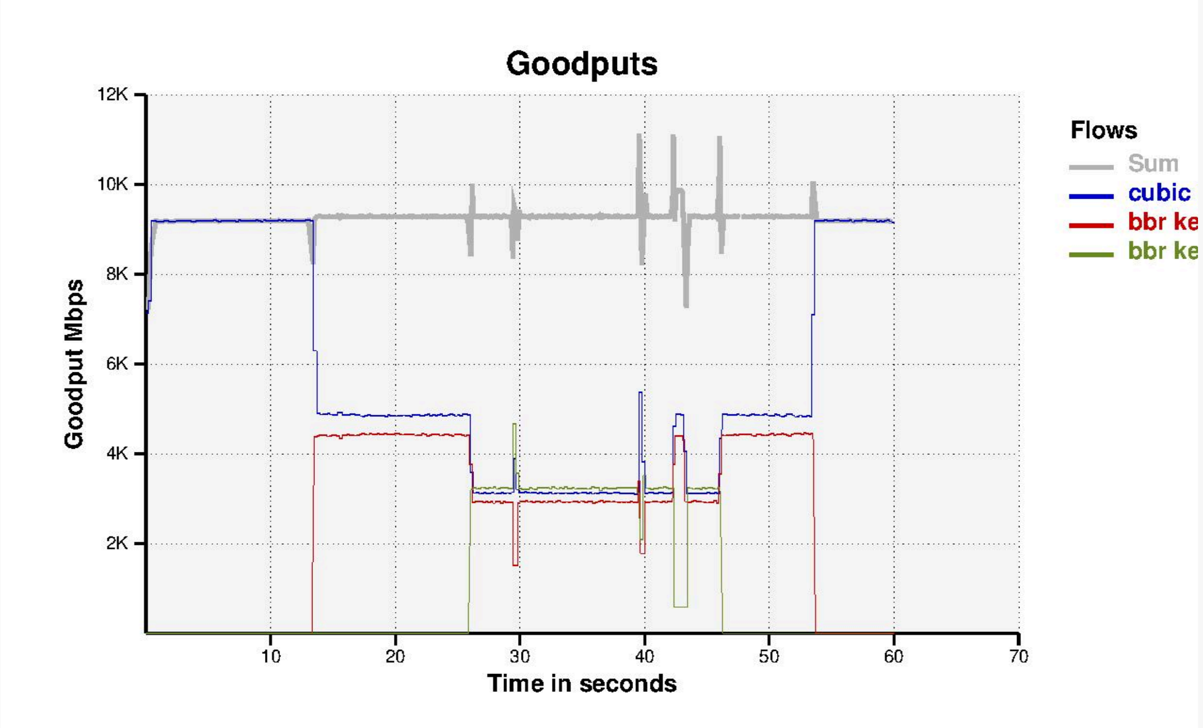


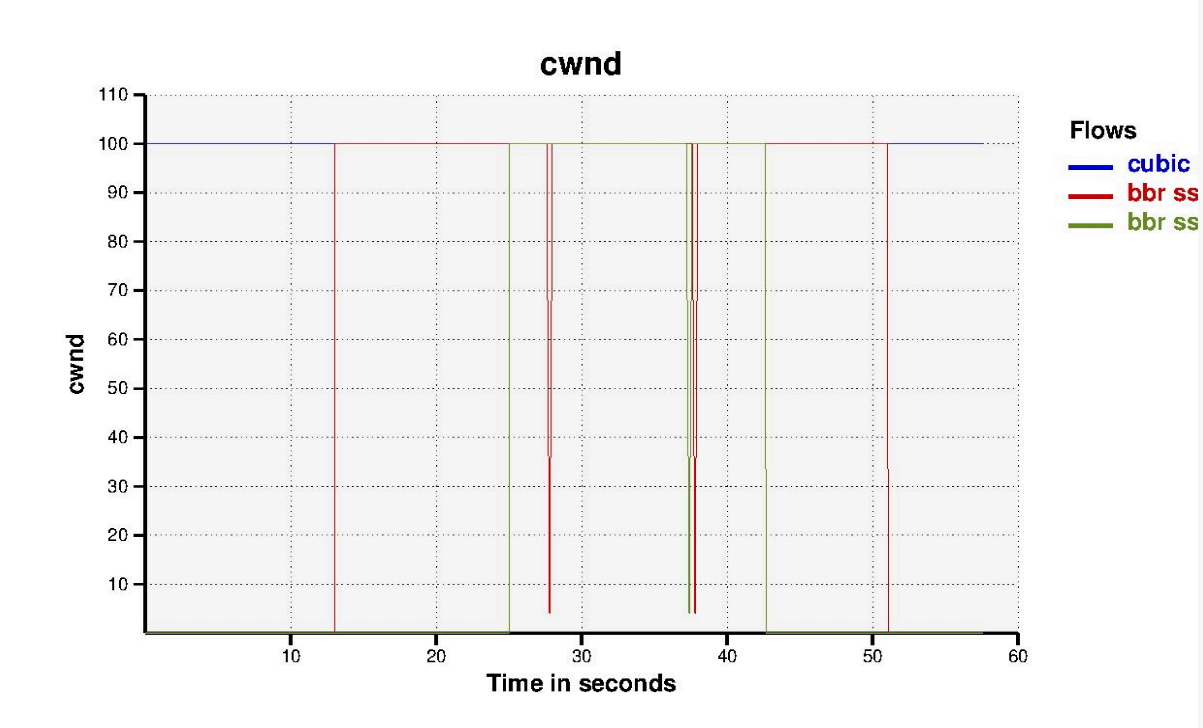
1-CUBIC VS. 2-DCTCP





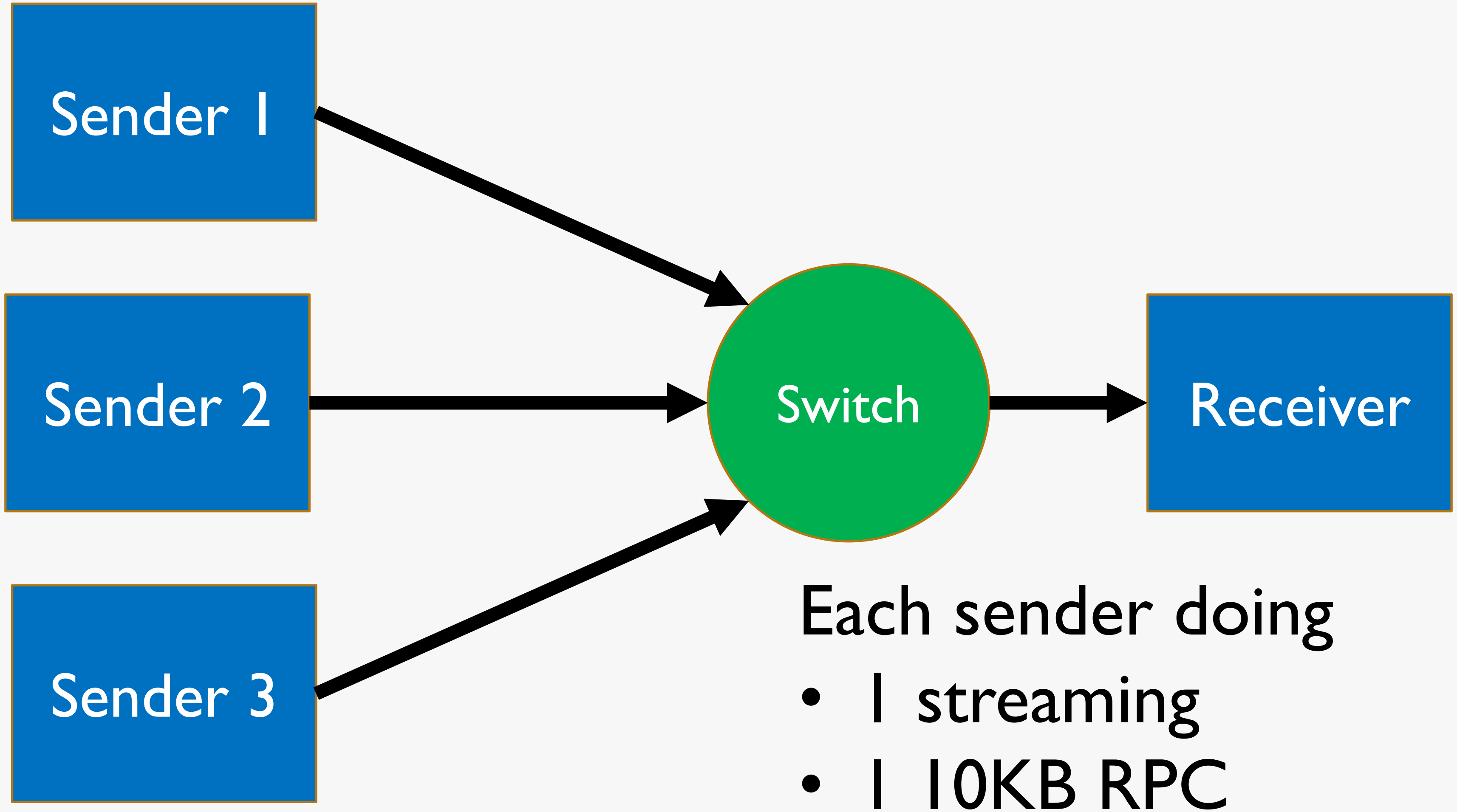
1-CUBIC VS. 2-BBR WITH TCP-BPF CLAMP





SIZE FAIRNESS AND MANY FLOWS

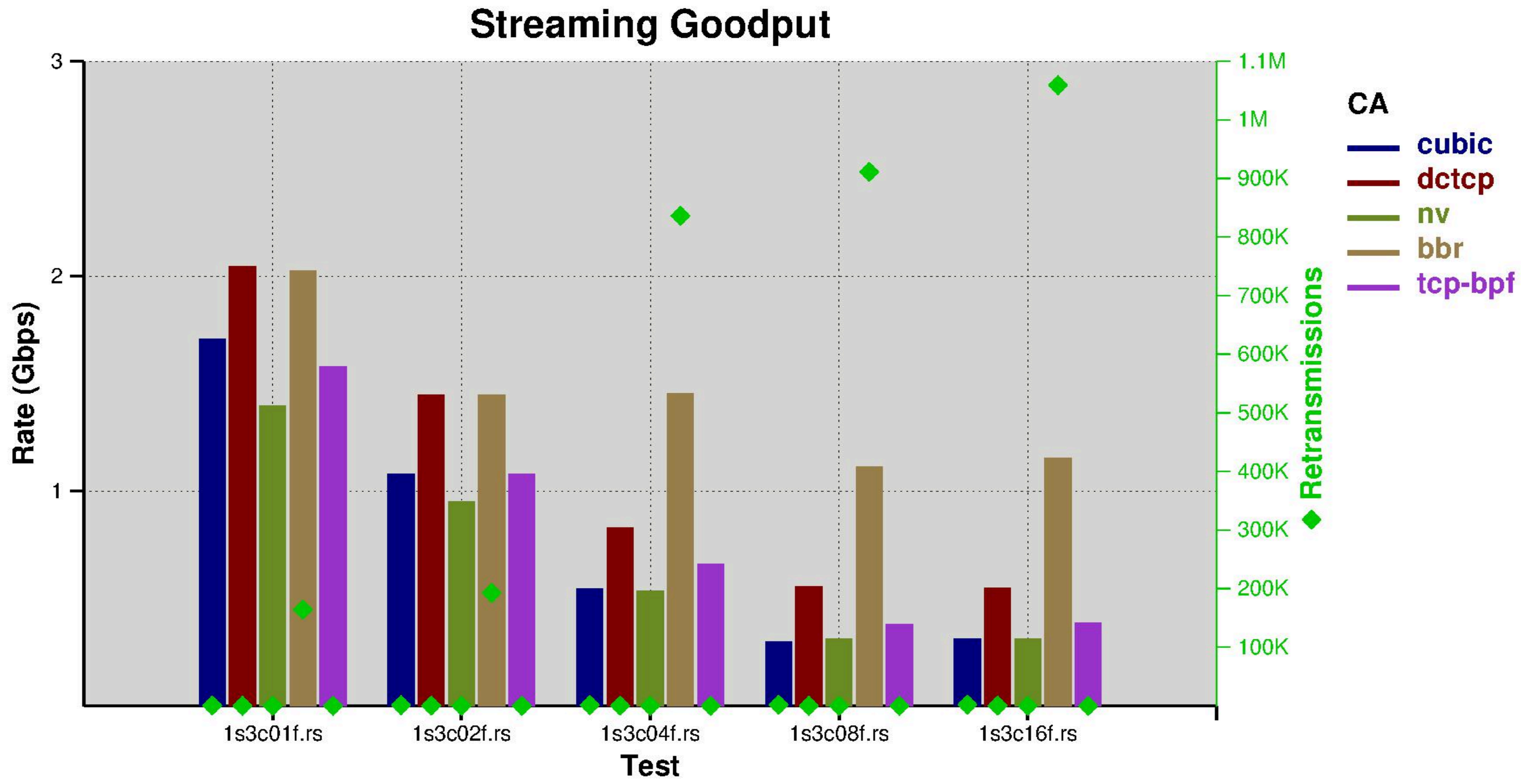
SIZE FAIRNESS AND MANY FLOWS



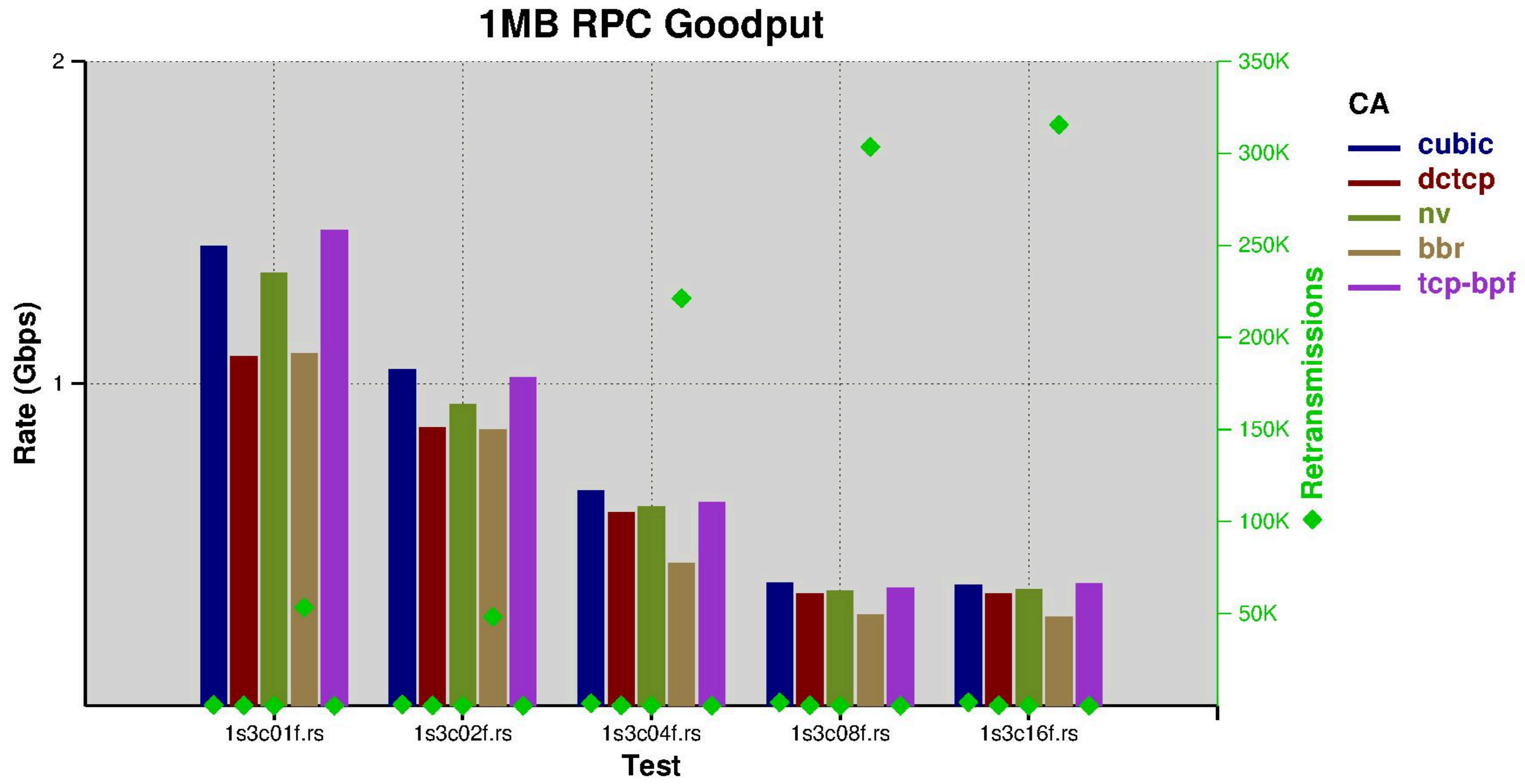
Each sender doing

- 1 streaming
- 1 10KB RPC
- *x* 1MB RPCs

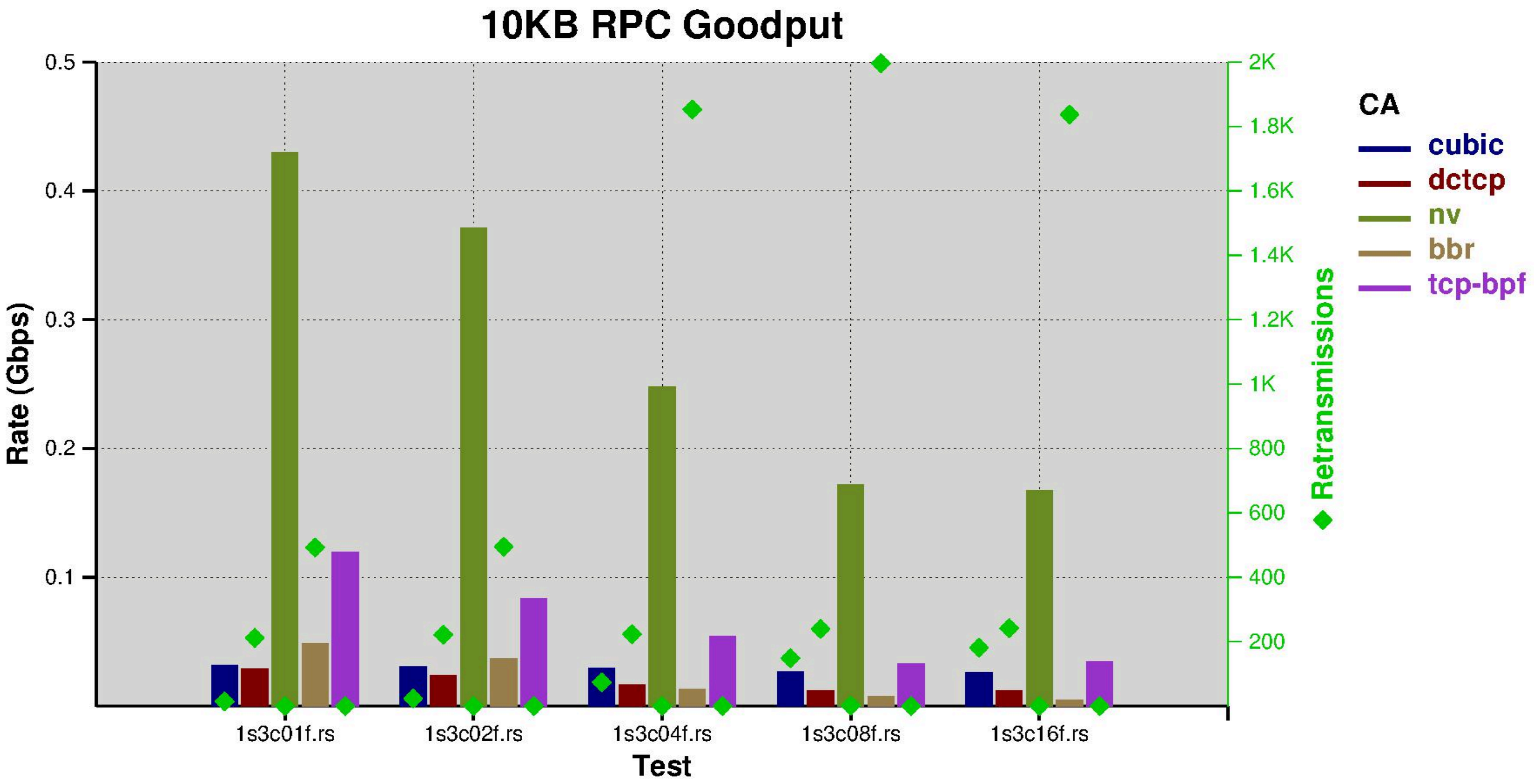
STREAMING GOODPUT AND RETRANSMISSIONS



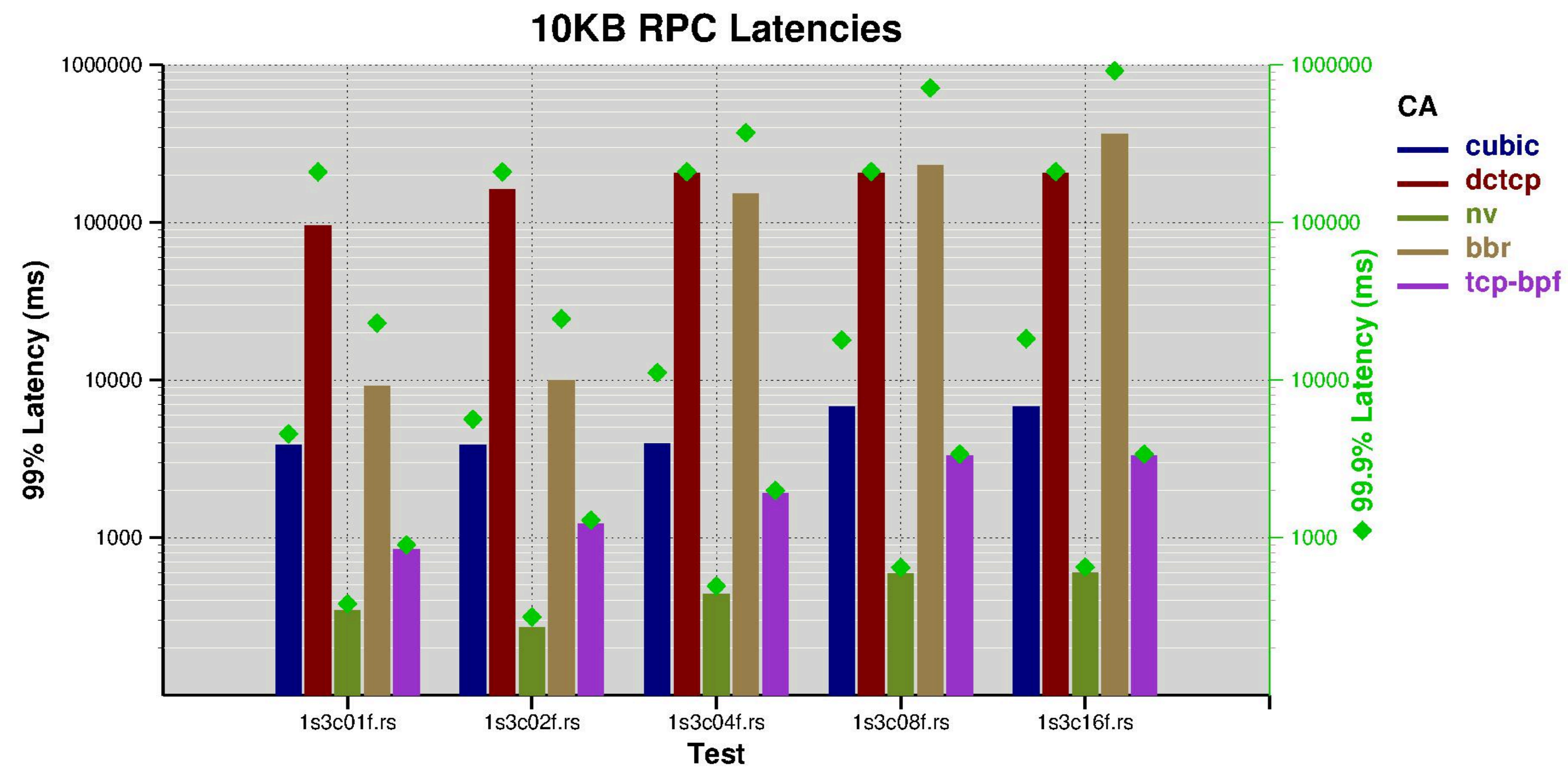
1MB RCP GOODPUTS AND RETRANSMISSIONS



10KB RPC GOODPUT AND RETRANSMISSIONS

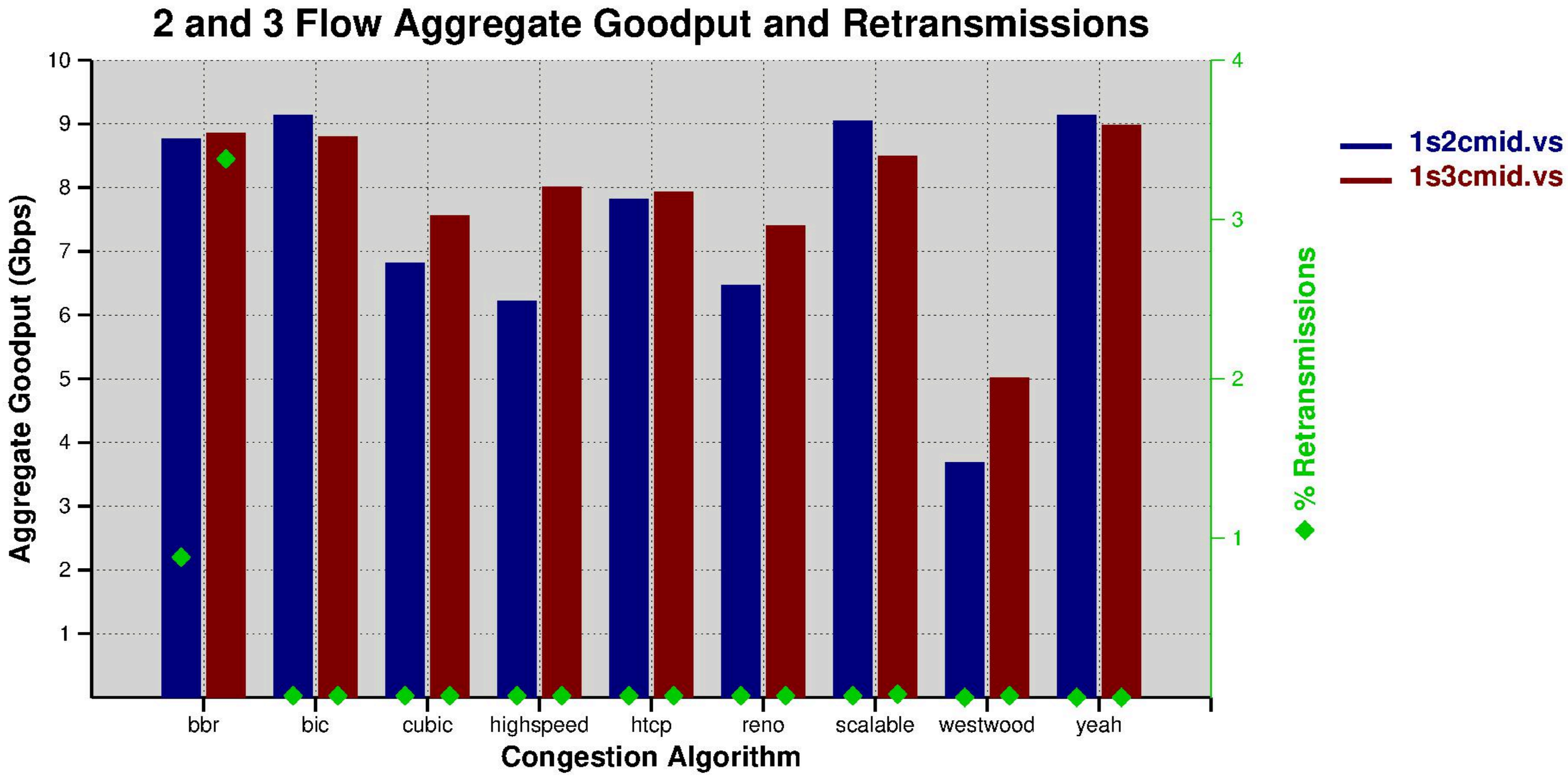


10KB RPC LATENCIES

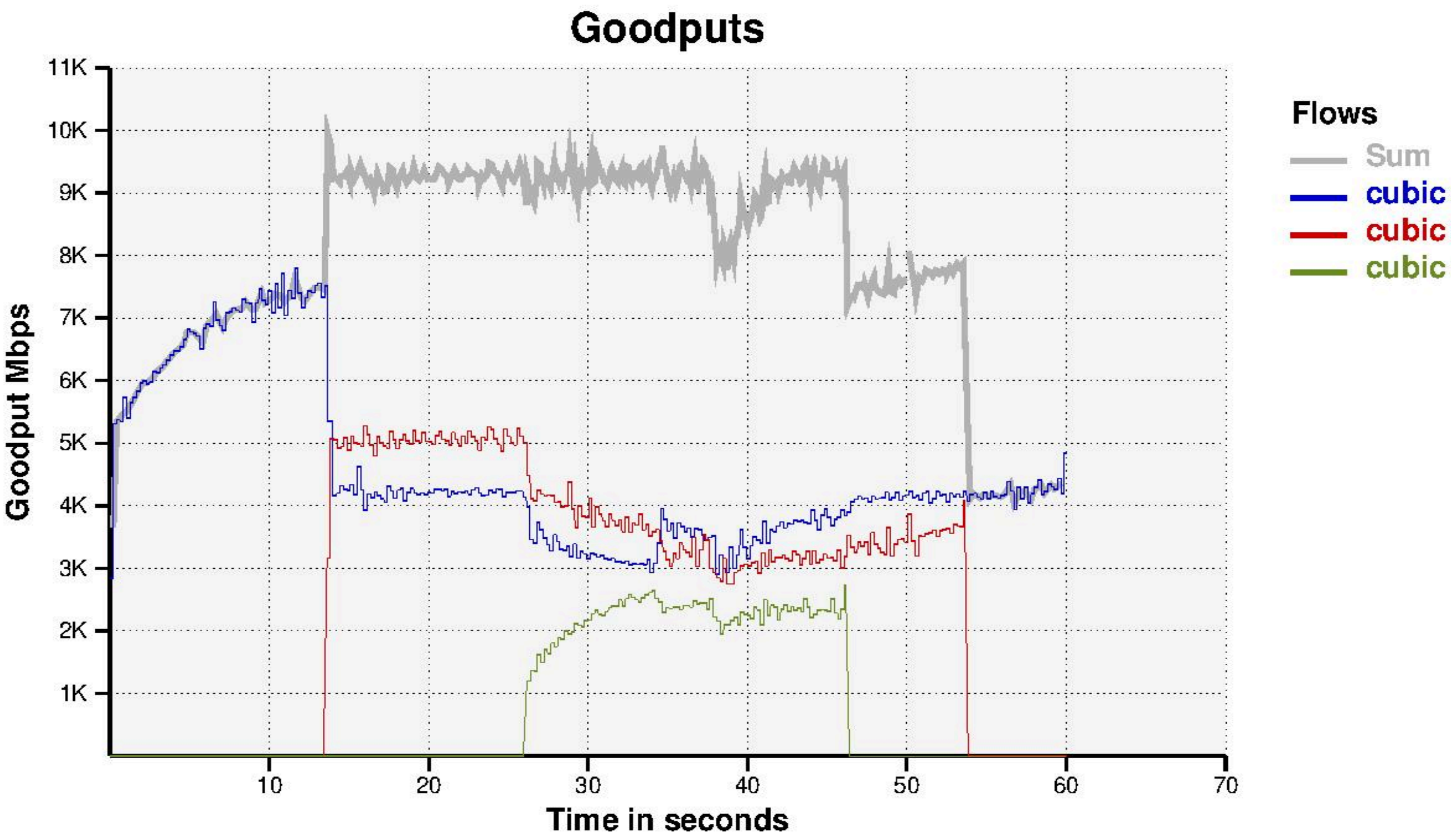


10G-10MS SCENARIOS

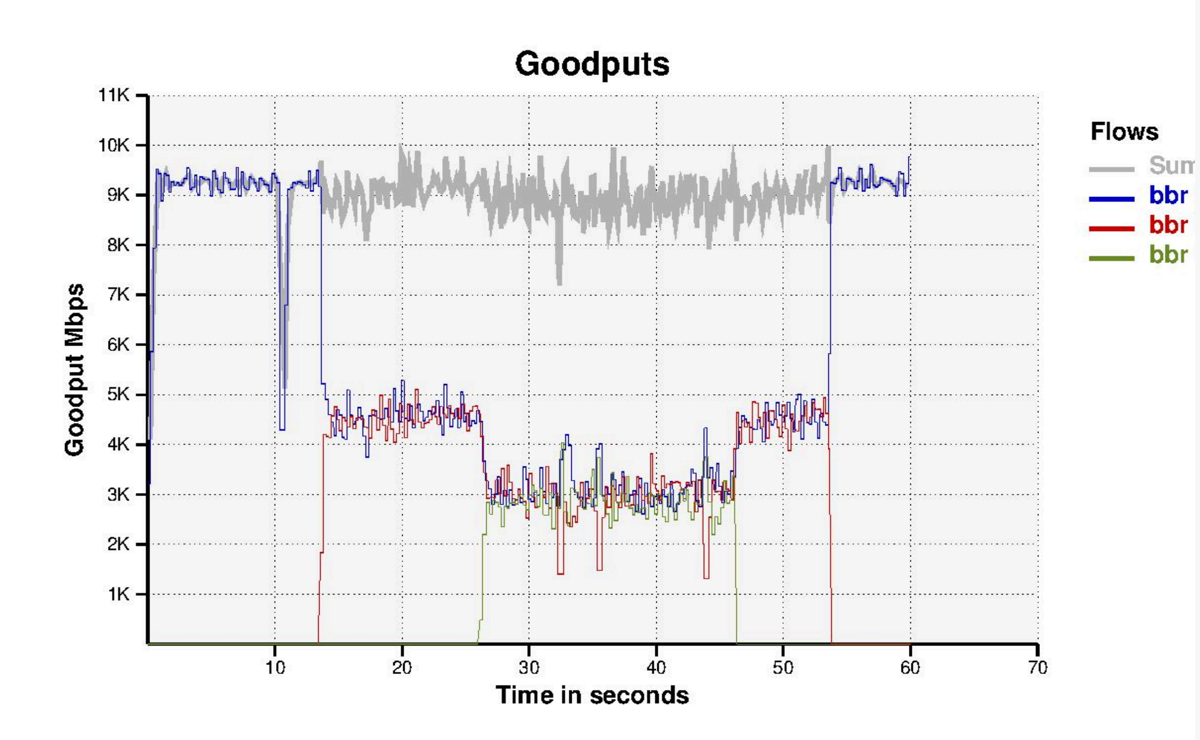
2 OR 3 FLOWS ALL SAME CA



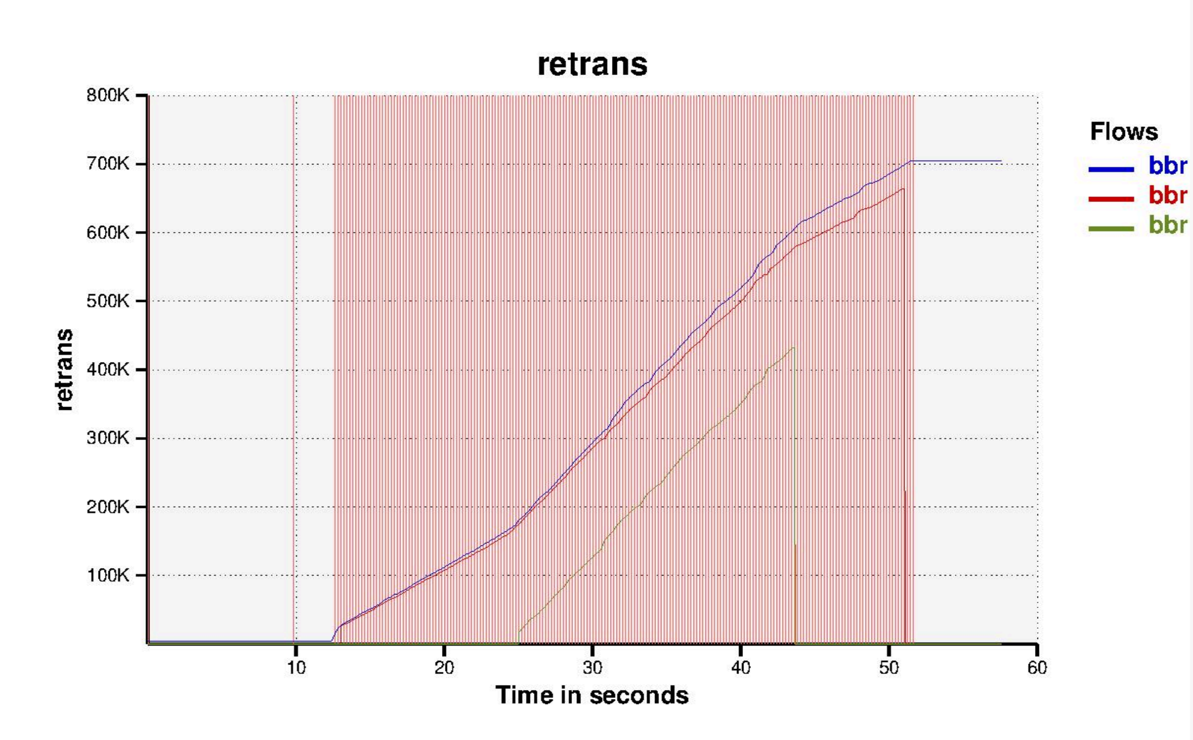
3 FLOW CUBIC


Goodputs
Goodput Mbps
11K
10K
9K
8K
7K
6K
5K
4K
3K
2K
1K
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
cubic
cubic
cubic

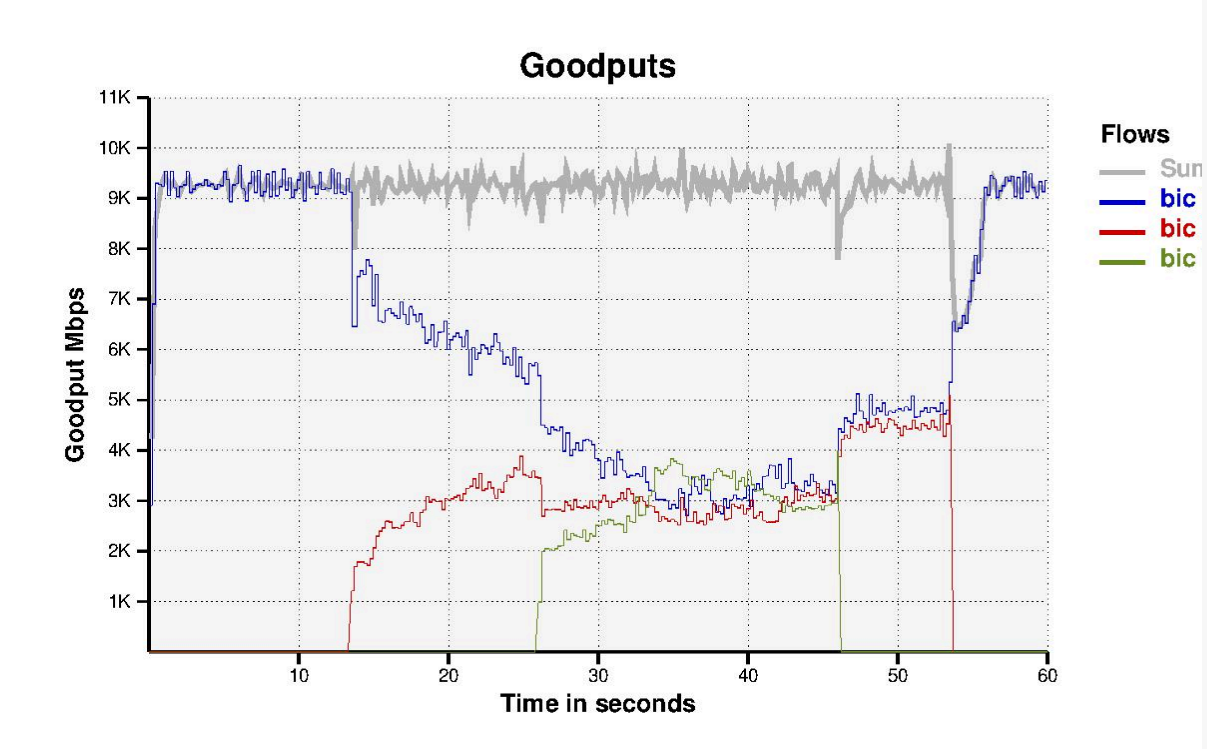
3-FLOW BBR

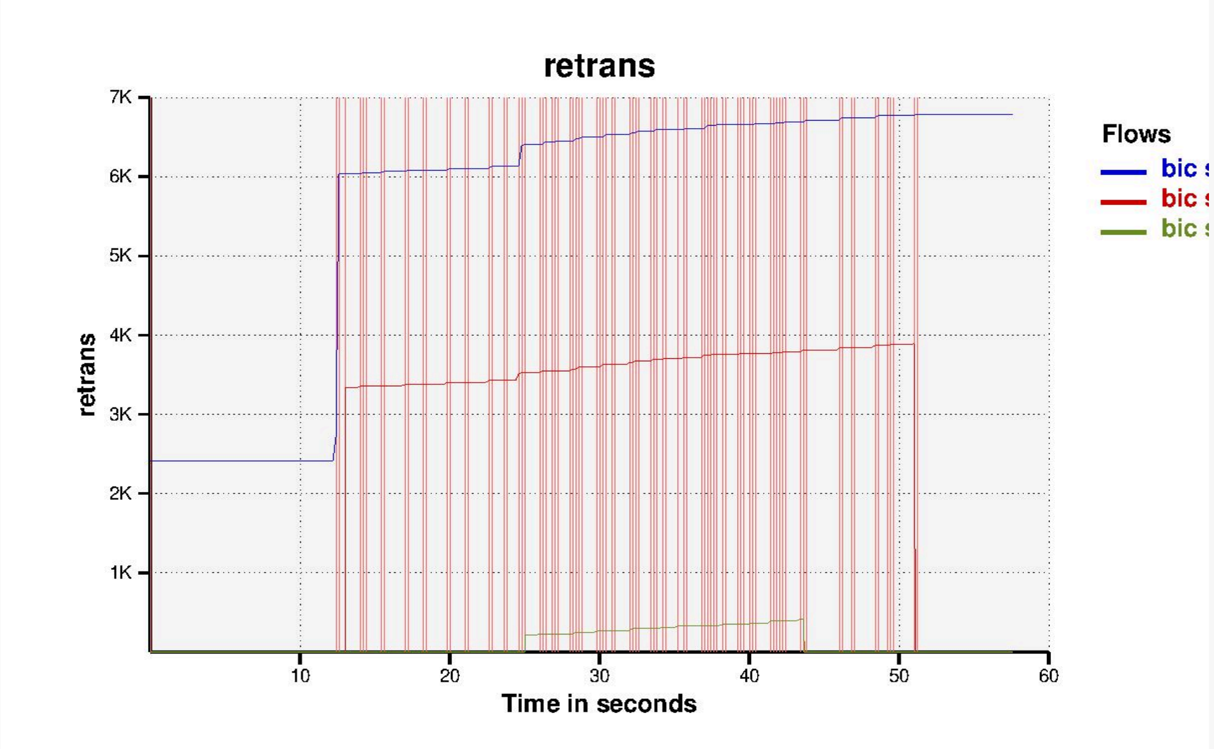


Goodputs
Goodput Mbps
11K
10K
9K
8K
7K
6K
5K
4K
3K
2K
1K
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
bbr
bbr
bbr

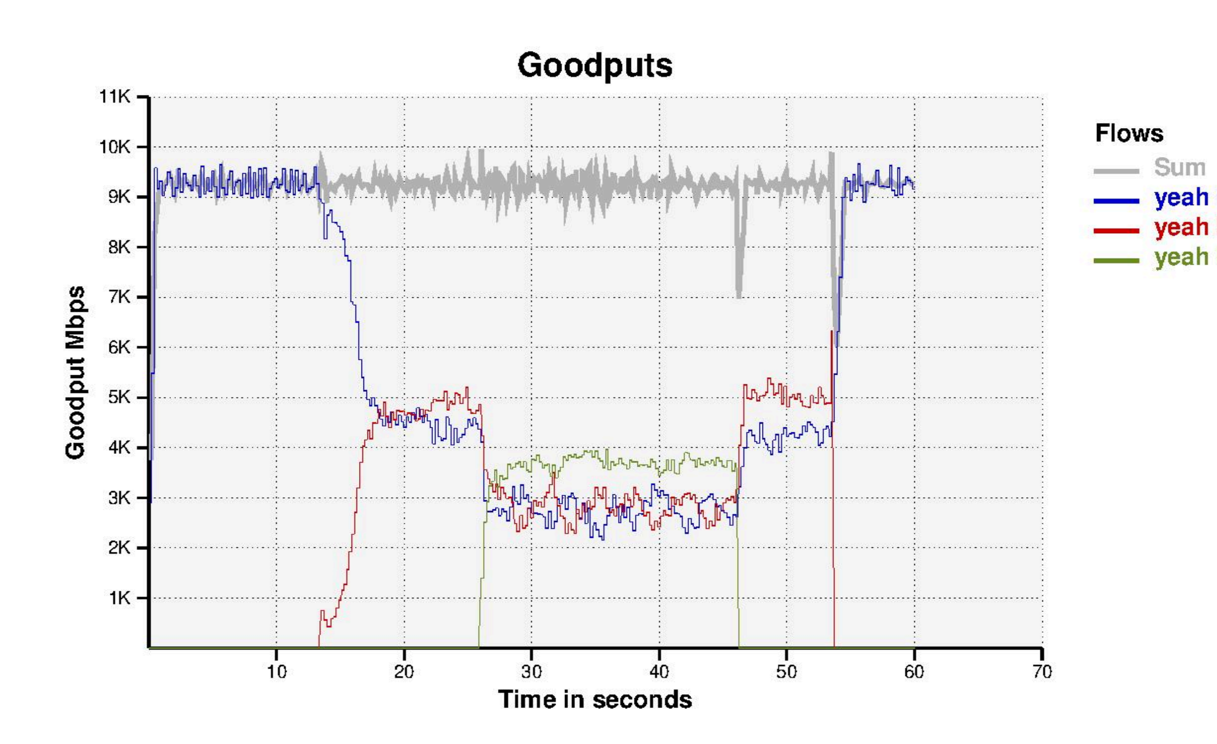


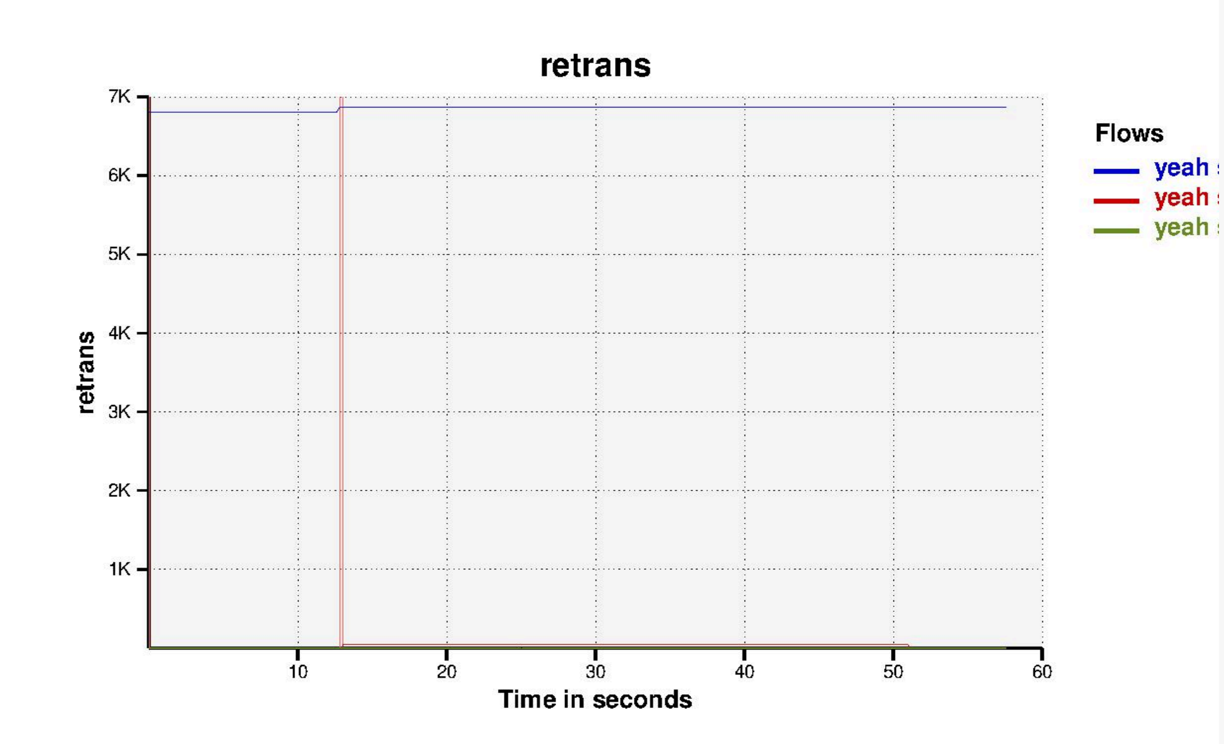
retrans
retrans
800K
700K
600K
500K
400K
300K
200K
100K
10
20
30
40
50
60
Time in seconds
Flows
bbr
bbr
bbr


Goodputs
11K
10K
9K
8K
7K
6K
5K
4K
3K
2K
1K
Goodput Mbps
10
20
30
40
50
60
Time in seconds
Flows
Sun
bic
bic
bic

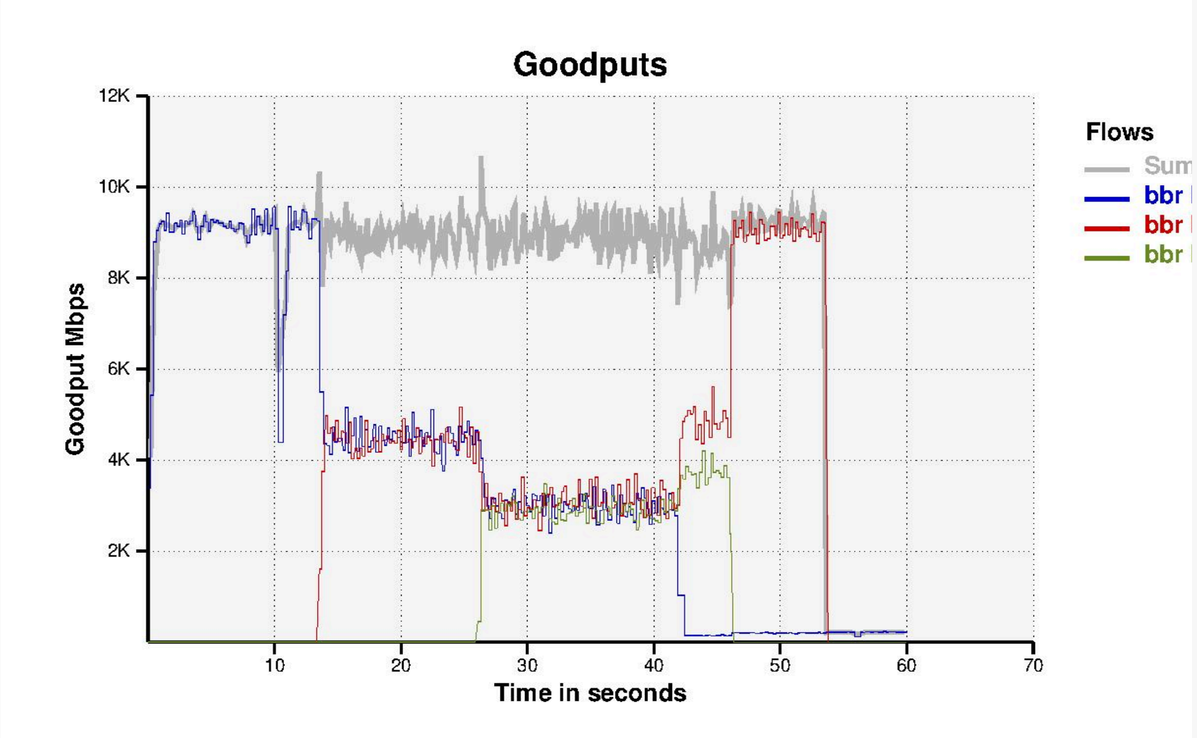

retrans
7K
6K
5K
4K
3K
2K
1K
retrans
10
20
30
40
50
60
Time in seconds
Flows
bic s
bic s
bic s

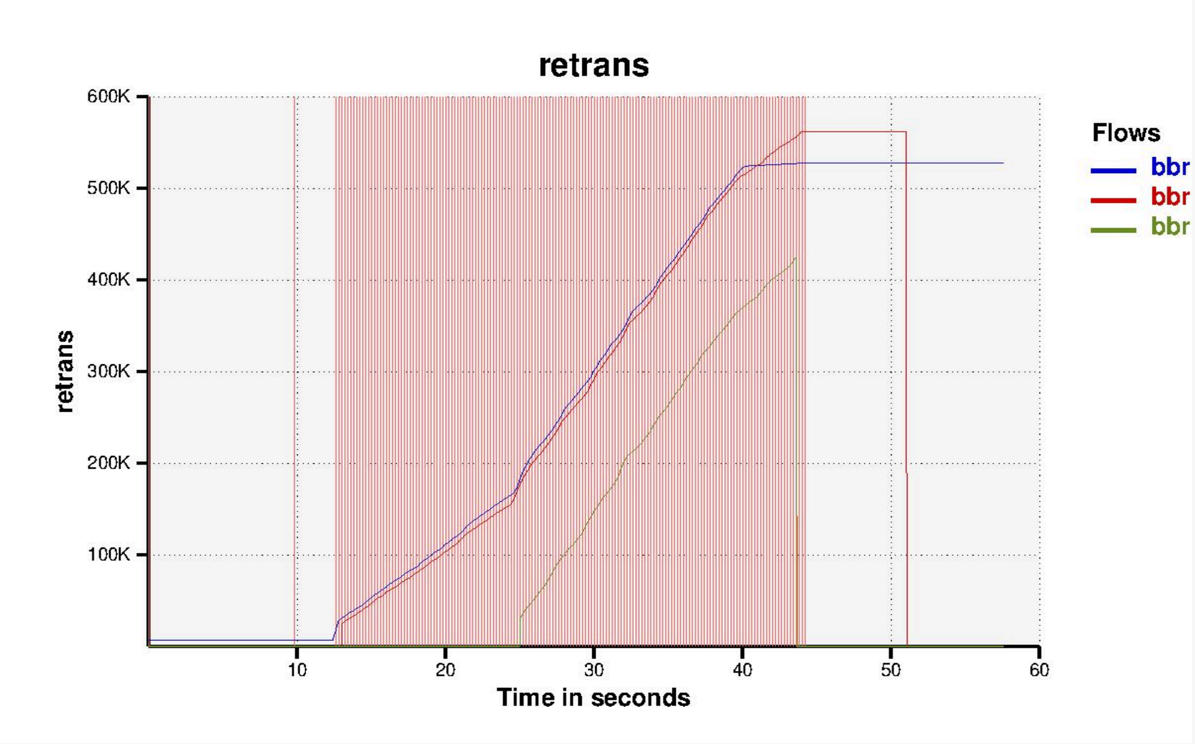
3 FLOW YEAH


Goodputs
Goodput Mbps
11K
10K
9K
8K
7K
6K
5K
4K
3K
2K
1K
10
20
30
40
50
60
70
Time in seconds
Flows
Sum
yeah
yeah
yeah

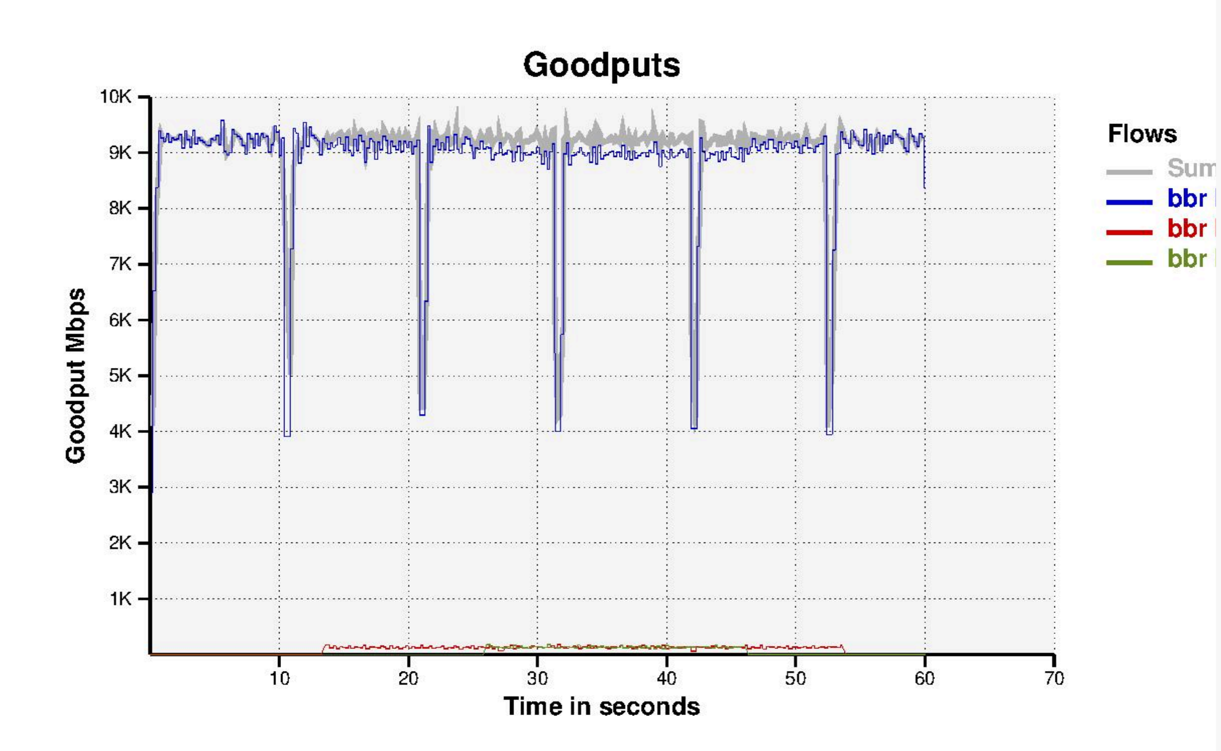

retrans
retrans
7K
6K
5K
4K
3K
2K
1K
10
20
30
40
50
60
Time in seconds
Flows
yeah
yeah
yeah

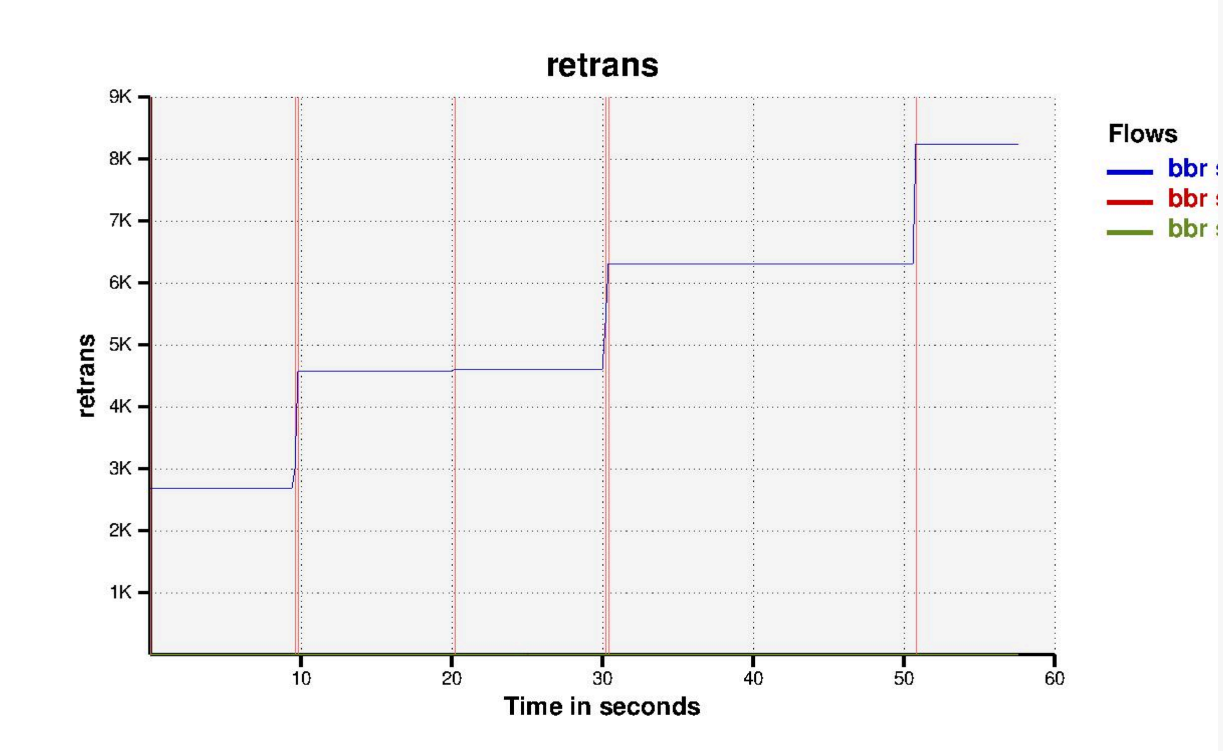
BBR FLOW COLLAPSE





3 FLOW BBR BAD COLLAPSE





BBR COLLAPSES

- Collapses seen in 10% of 2 flow BBR tests
- Collapses seen in 20% of 3 flow BBR tests

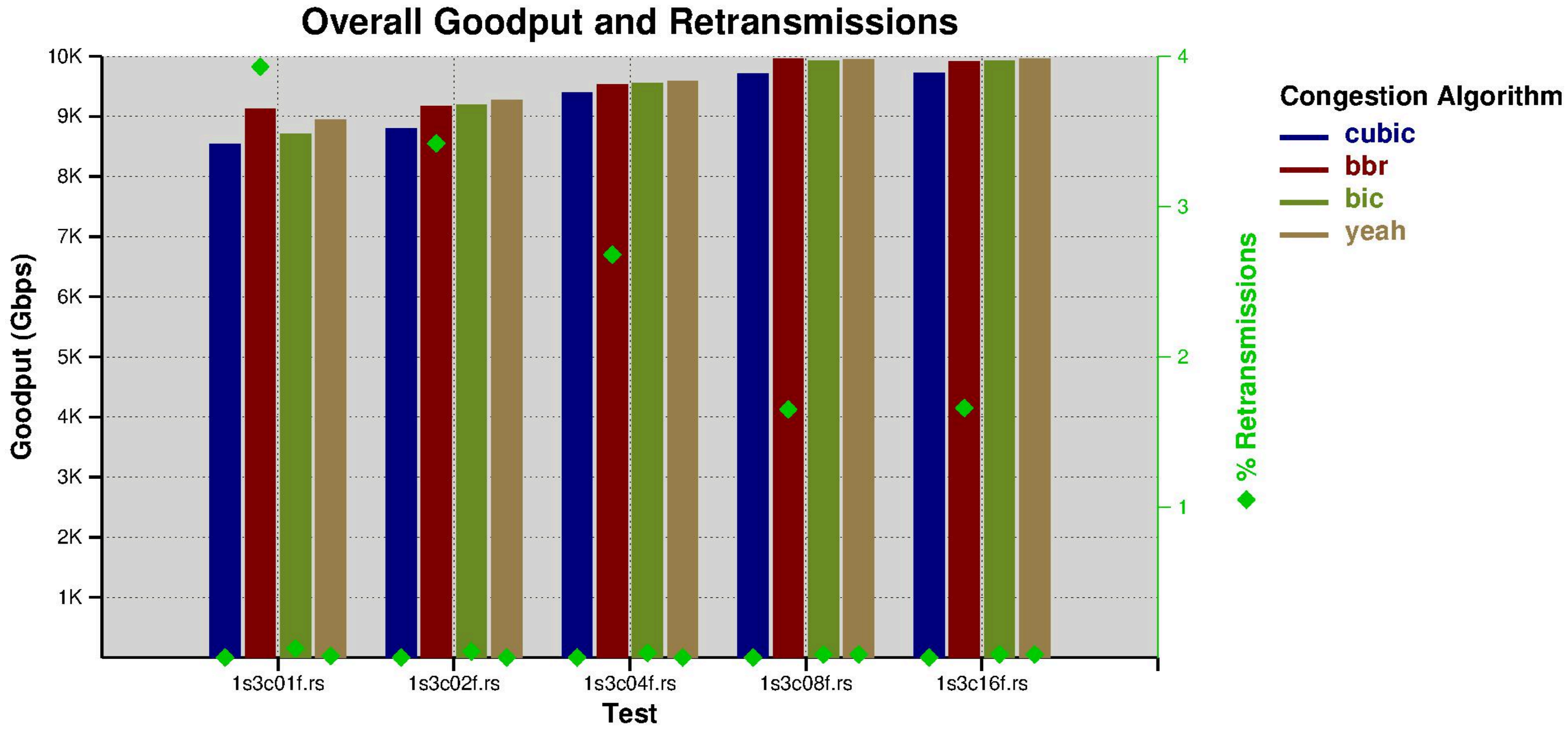
- It is possible that some netem is causing collapse, but
  - Other people have seen it without using netem
  - It should be able to recover

FAIRNESS AGAINST CUBIC

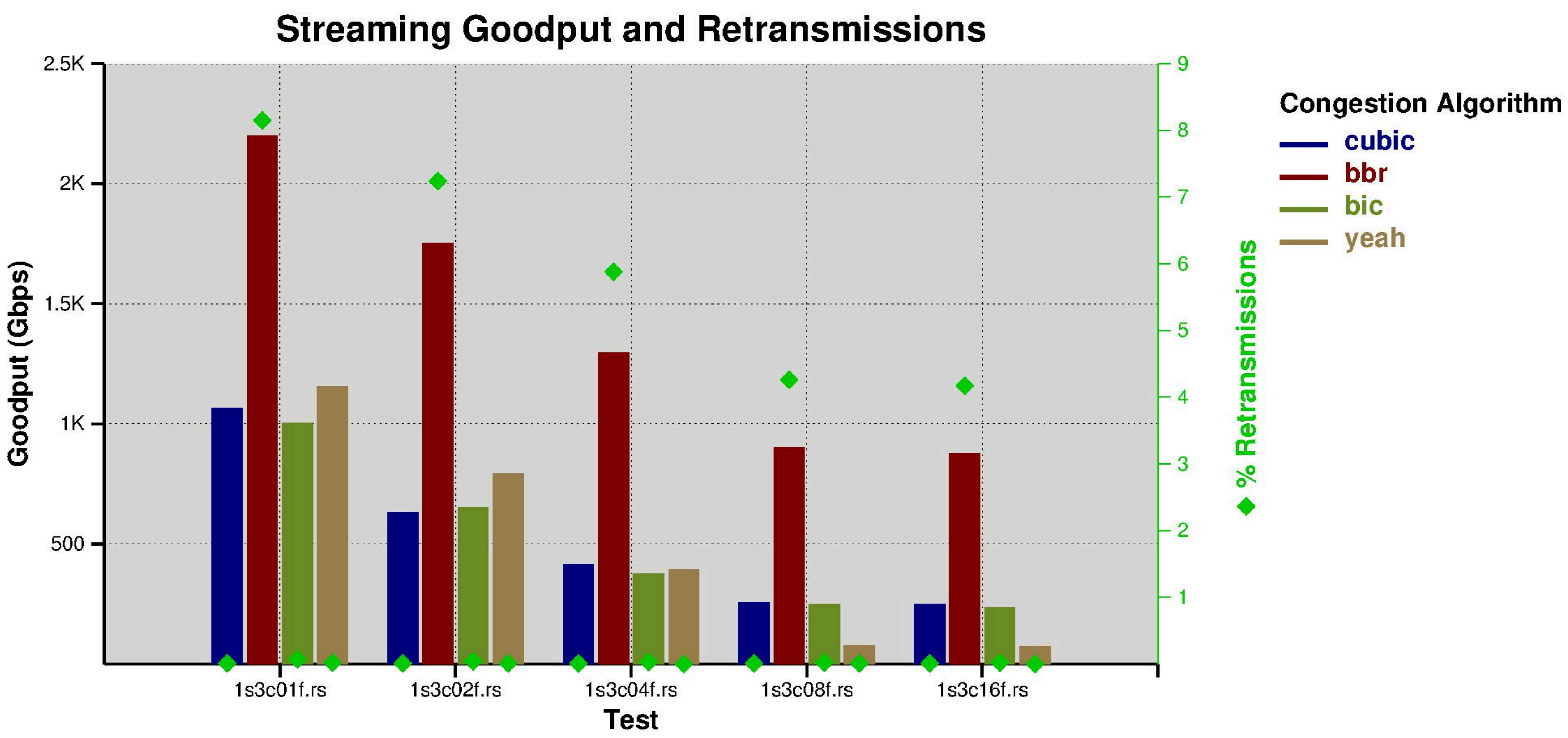
- Cubic looses against BIC and BBR
- Yeah losses against Cubic
- Cubic and Reno even

SIZE FAIRNESS AND MANY FLOWS

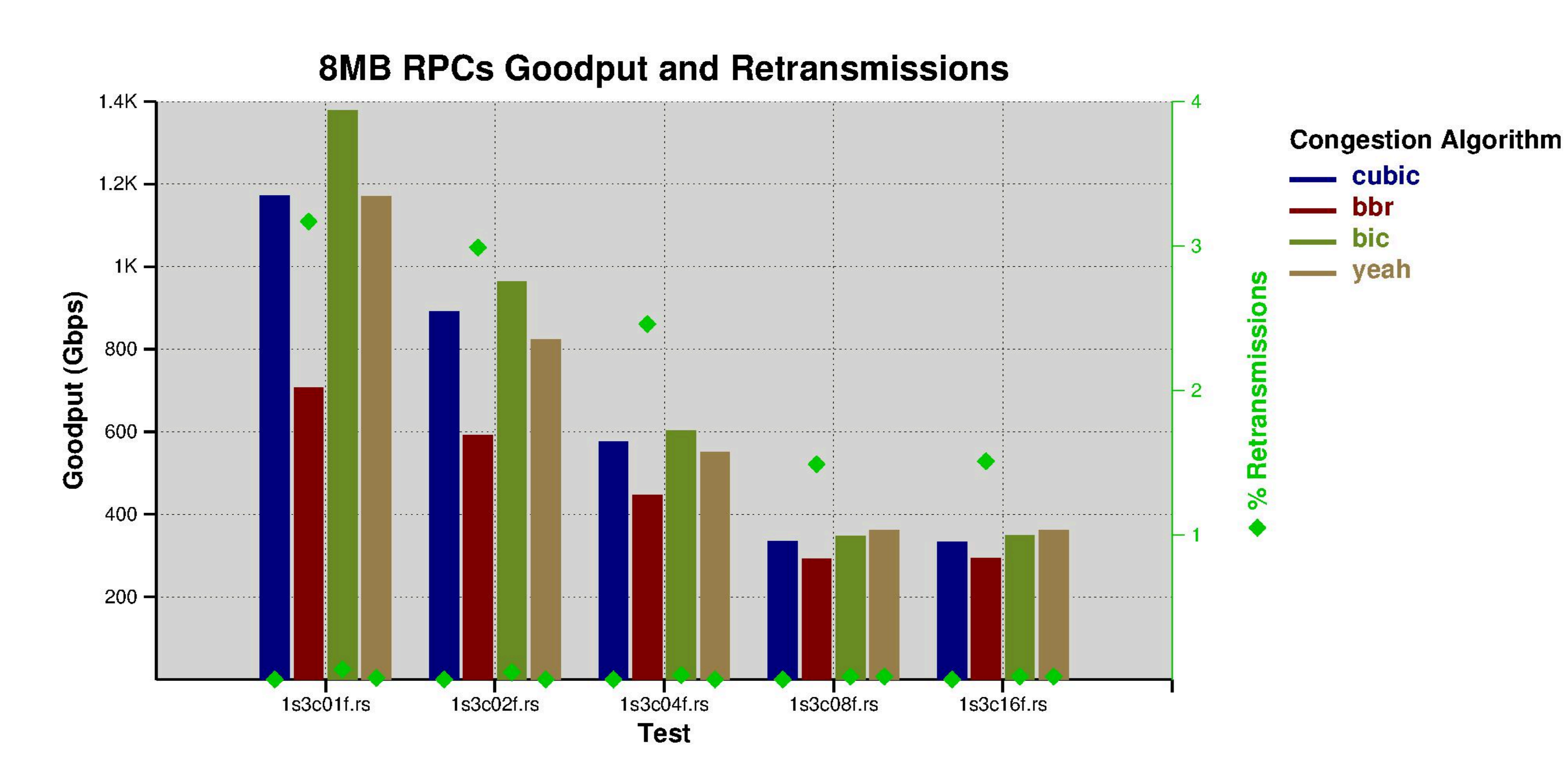
OVERALL GOODPUT AND RETRANSMISSIONS



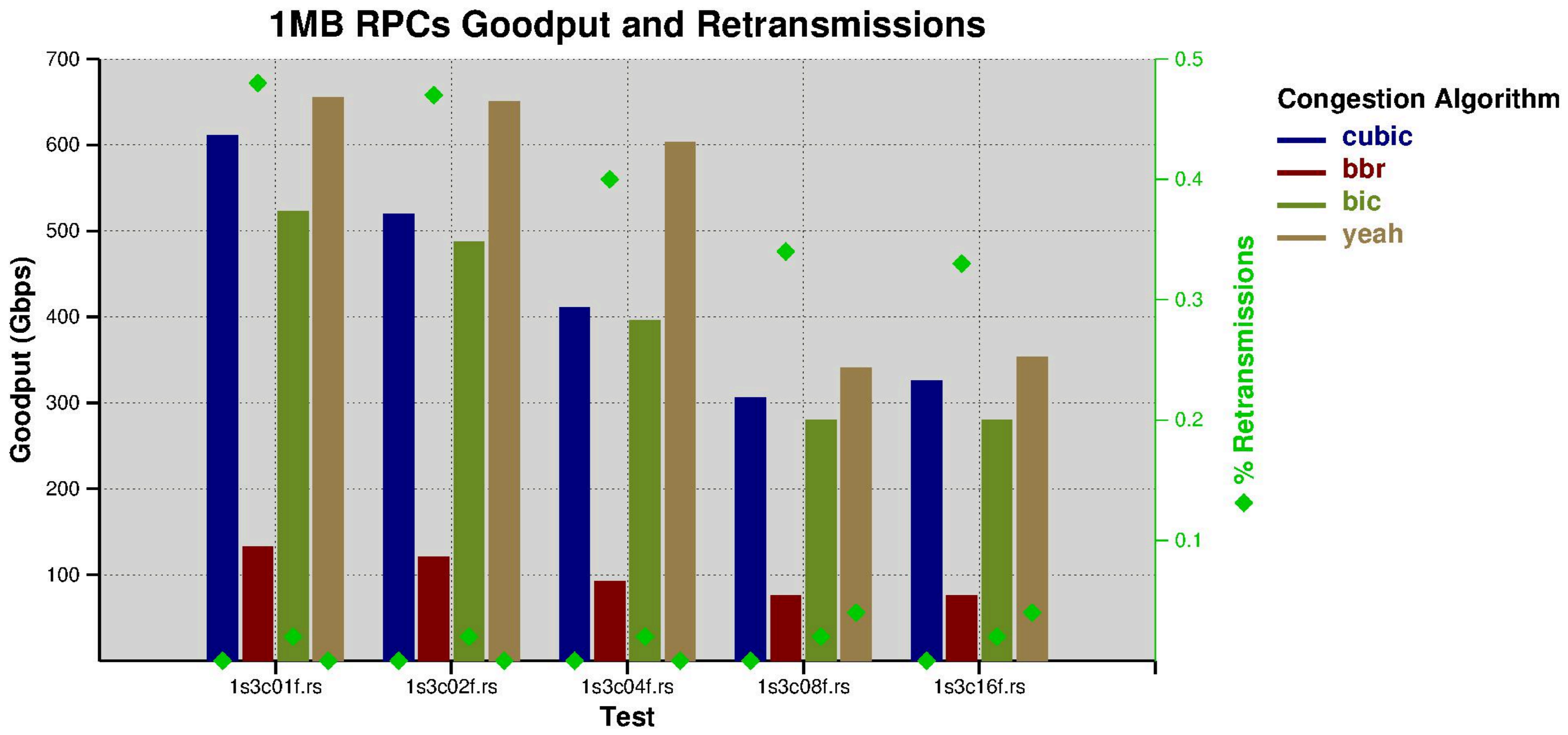
STREAMING GOODPUT



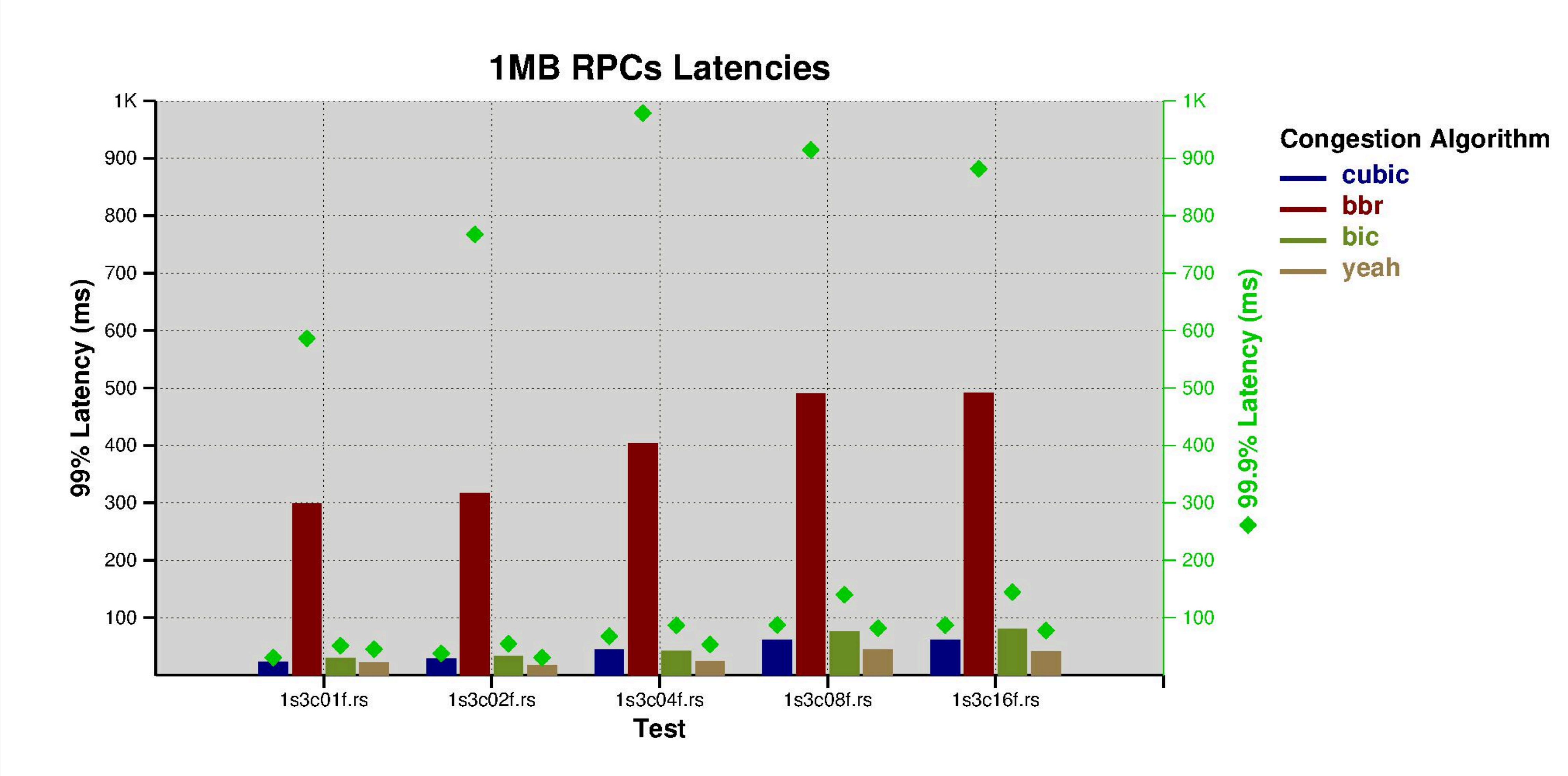
8MB GOODPUT


8MB RPCs Goodput and Retransmissions
Goodput (Gbps)
1.4K
1.2K
1K
800
600
400
200
% Retransmissions
4
3
2
1
1s3c01f.rs
1s3c02f.rs
1s3c04f.rs
1s3c08f.rs
1s3c16f.rs
Test
Congestion Algorithm
cubic
bbr
bic
yeah

1MB GOODPUT


1MB RPCs Goodput and Retransmissions
Goodput (Gbps)
700
600
500
400
300
200
100
% Retransmissions
0.5
0.4
0.3
0.2
0.1
Test
1s3c01f.rs
1s3c02f.rs
1s3c04f.rs
1s3c08f.rs
1s3c16f.rs
Congestion Algorithm
cubic
bbr
bic
yeah

1MB RPC LATENCIES

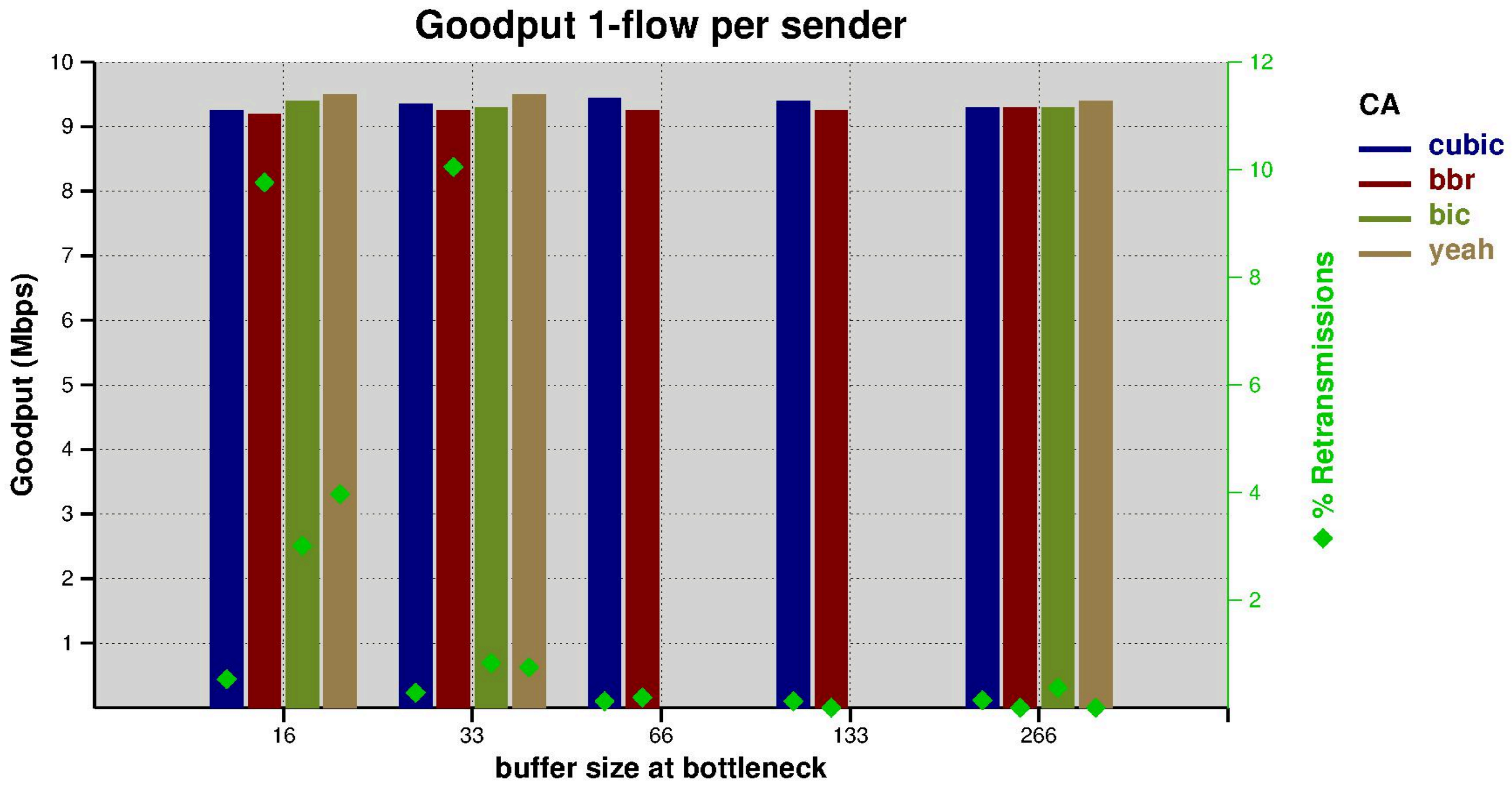


10G-10MS RESULTS

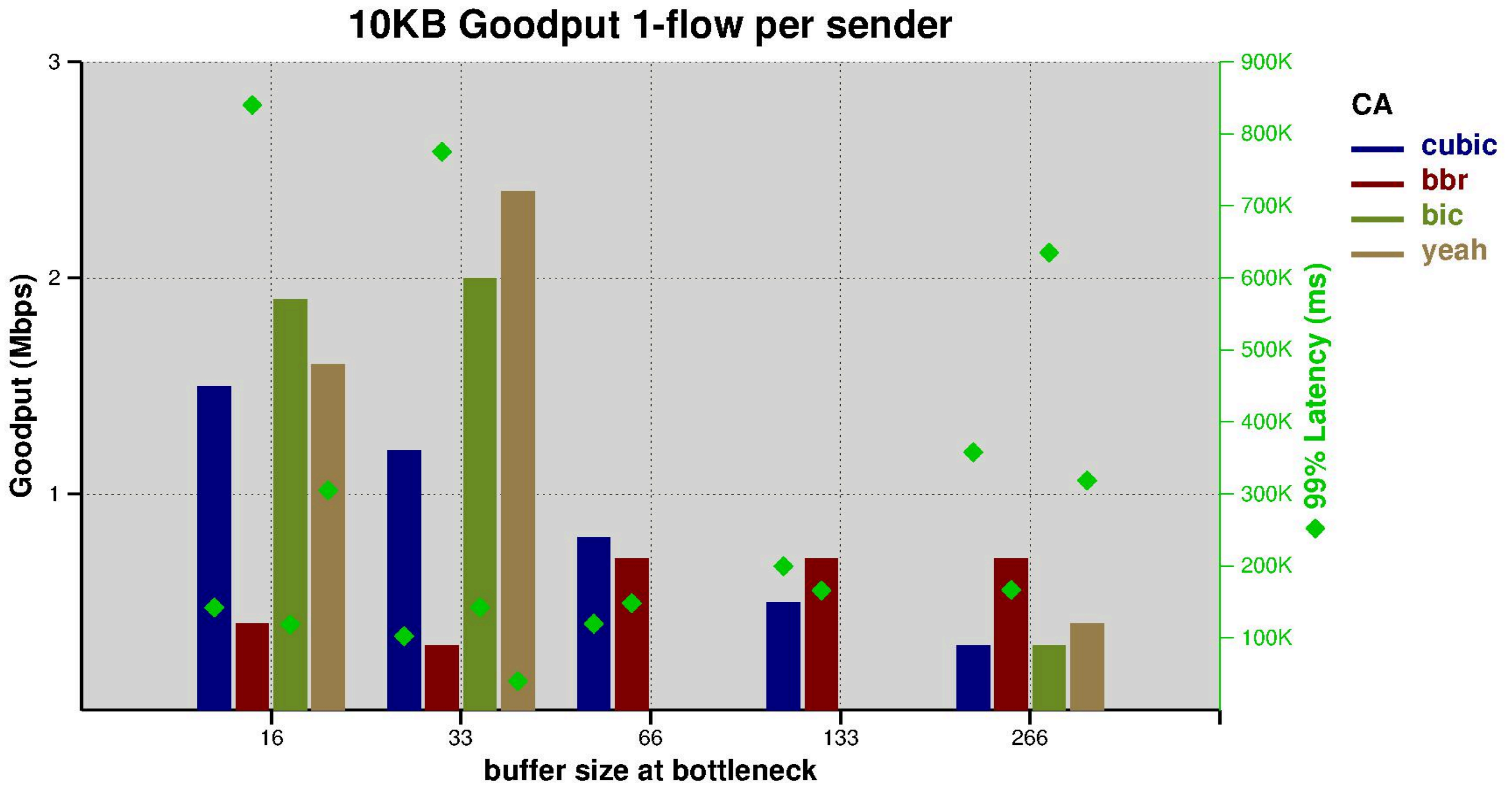
- No CA is perfect
- Yeah suffers against Cubic
- BBR and BIC hurt Cubic
- BBR is good at using available bandwidth
- BBR does well when it is the only flow
- BBR hurts itself
- BBR has a lot of retransmissions

40MS RTT, 10 MBITS/S

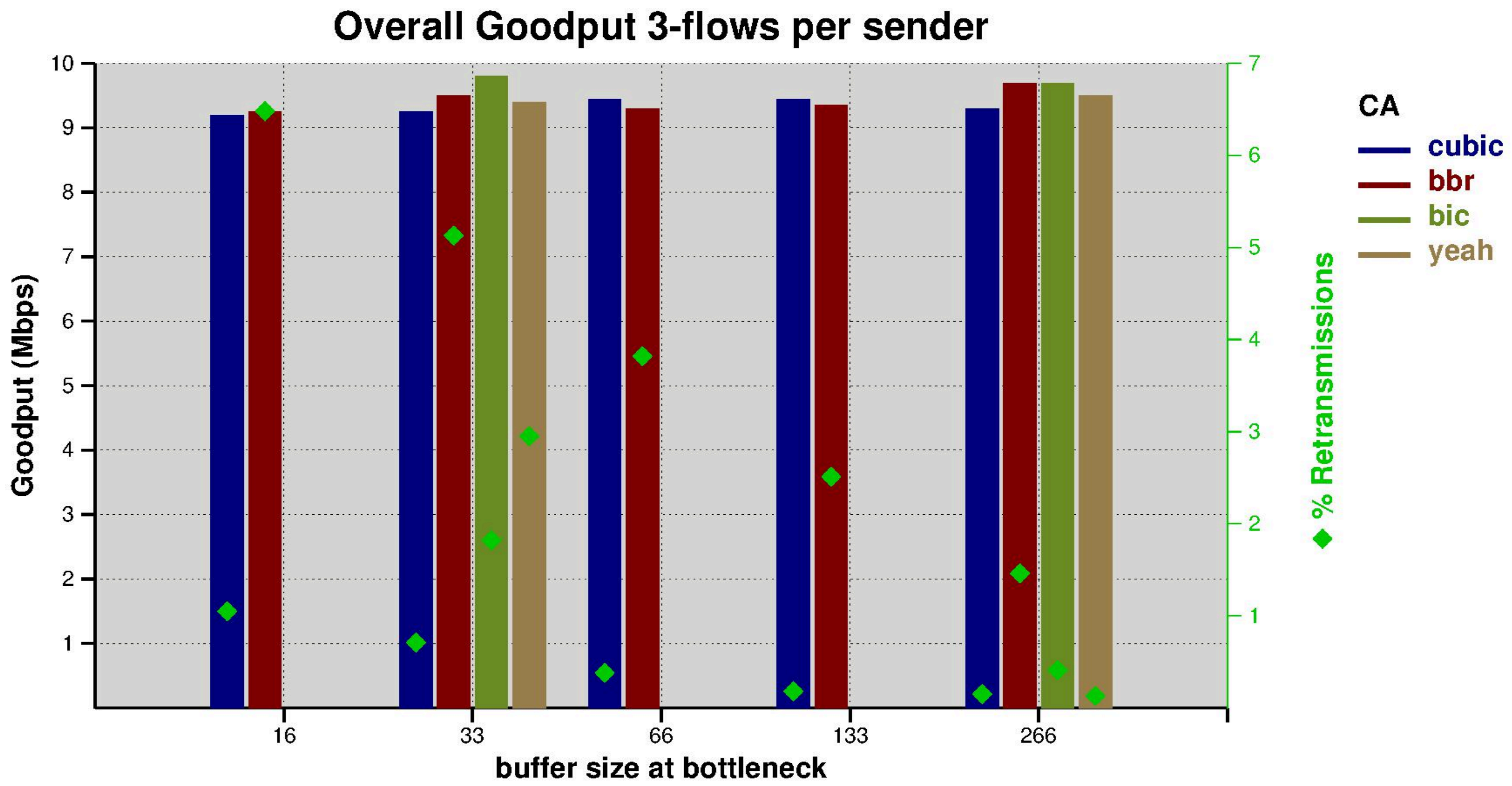
1-FLOW PER HOST, OVERALL GOODPUT



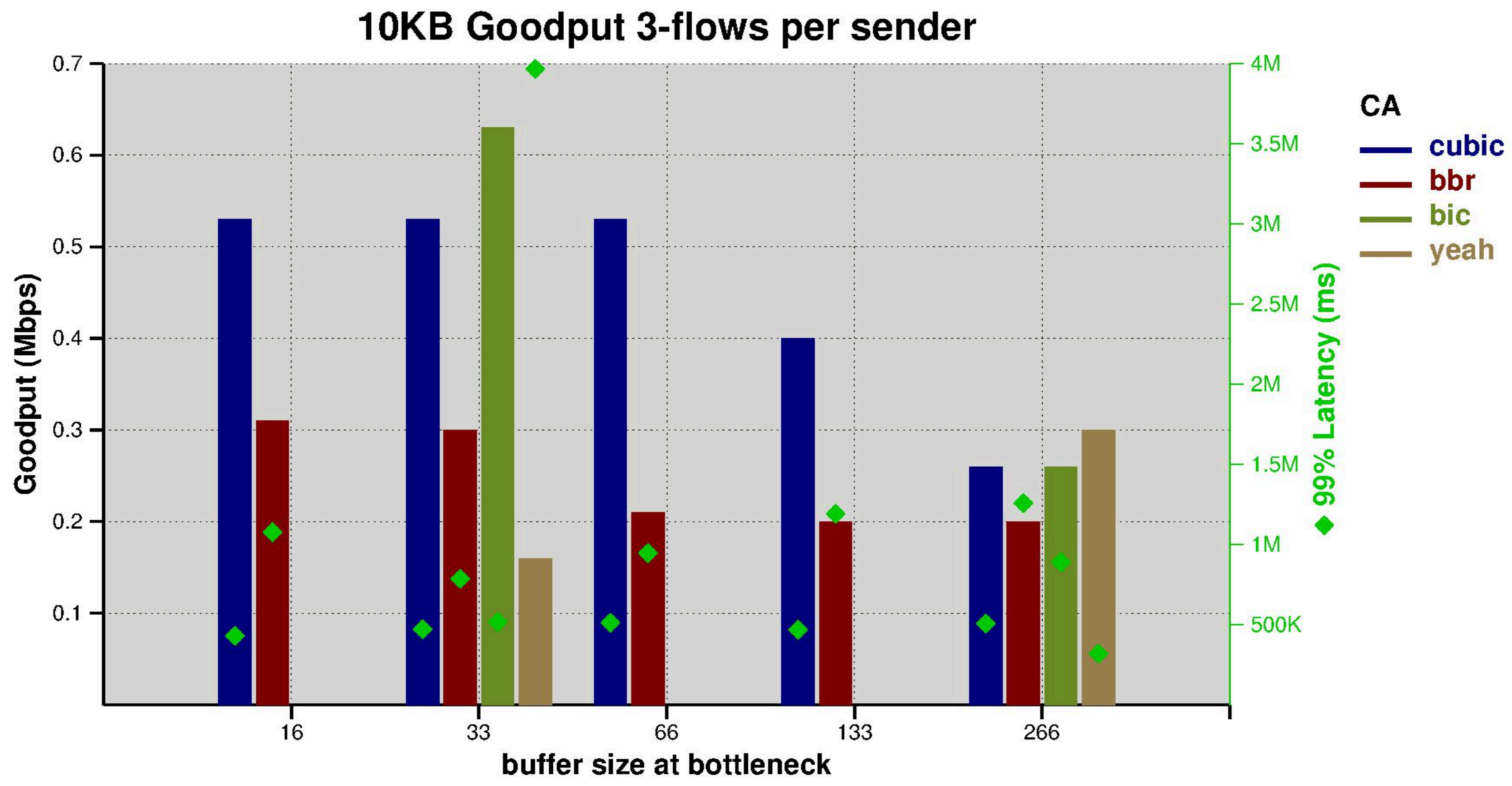
1-FLOW PER HOST, 10KB GOODPUT



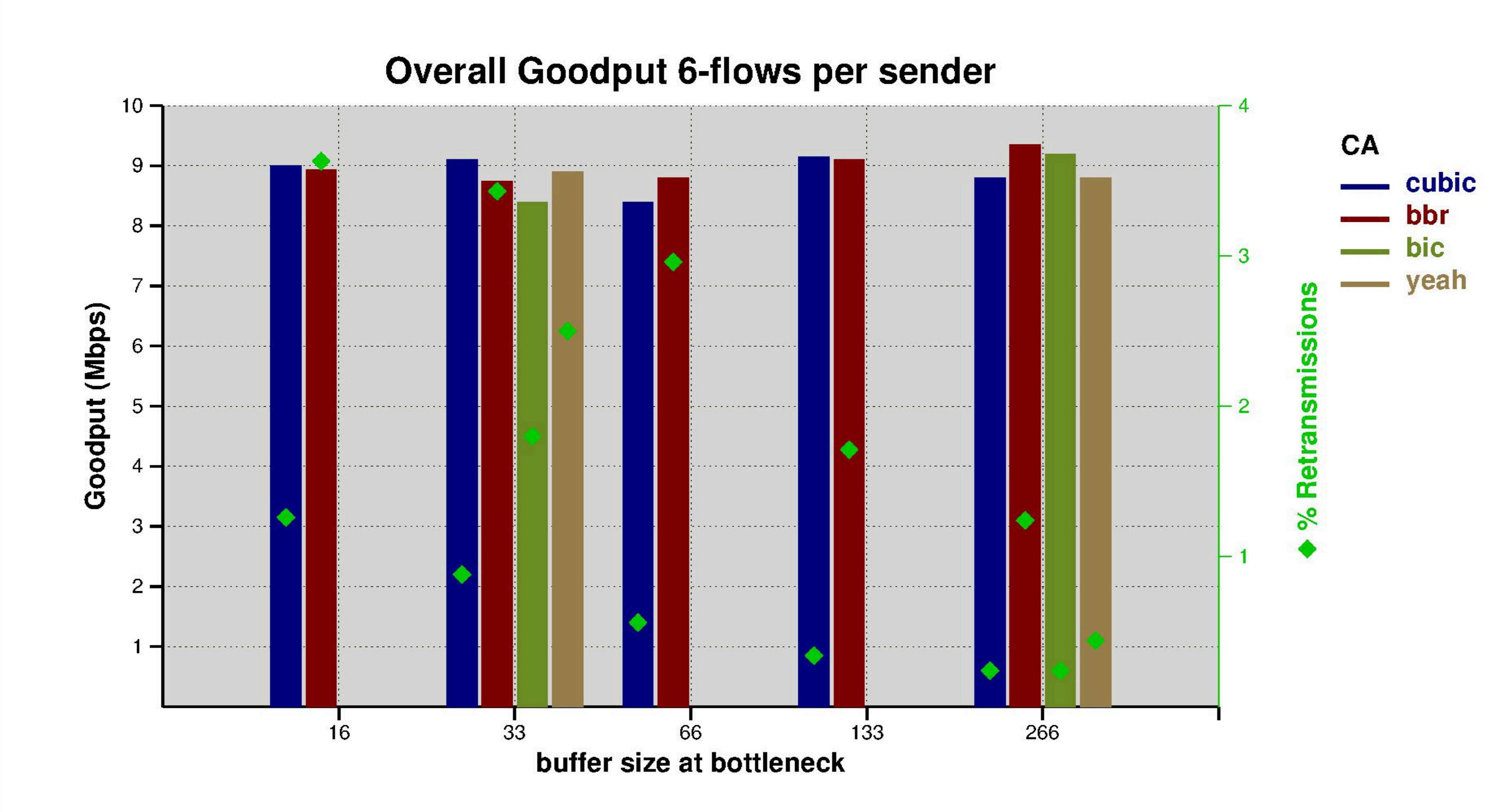
3-FLOW PER HOST, OVERALL GOODPUT



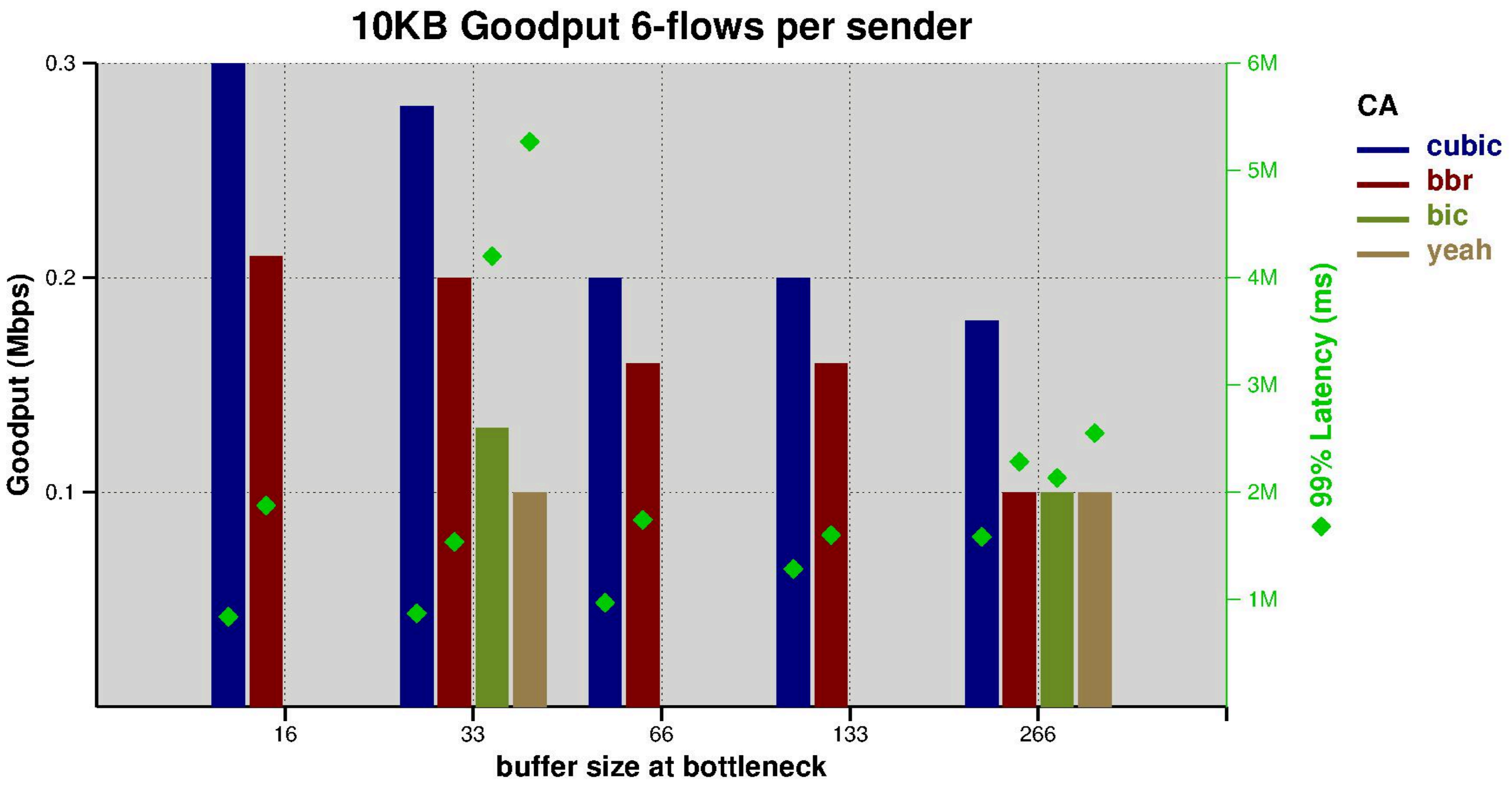
3-FLOW PER HOST, 10KB GOODPUT



6-FLOW PER HOST, OVERALL GOODPUT



6-FLOW PER HOST, 10KB GOODPUT


10KB Goodput 6-flows per sender
Goodput (Mbps)
buffer size at bottleneck
◆ 99% Latency (ms)
6M
5M
4M
3M
2M
1M
CA
cubic
bbr
bic
yeah