Testing FAST TCP over Abilene

Stanislav Shalunov ⟨shalunov@internet2.edu⟩

2nd PFLDnet Workshop, Argonne, 2004-02-17

Test of FAST TCP over Abilene

- Production circuit (2.5 Gb/s) saturated for 30 minutes
  - The circuit from the Abilene core T-640 router in Atlanta (ATLA) to the SoX GigaPoP
- No use of QoS techniques
- No fancy AQM (default drop-tail discipline was used)
- No adverse effects on conventional traffic
  - No losses
  - An increase of delay by relatively stable 5–6 ms

Test Sites: Five Throughput Continental US

SoX Southern Crossroads GigaPoP in Atlanta

PNW Pacific Northwest GigaPoP in Seattle

PSC Pittsburgh Supercomputing Center in Pittsburgh

NC-ITEC North Carolina Internet2 Technology Evaluation Center in Raleigh

STTL The measurement rack in a Qwest PoP next to the Abilene core router in Seattle

Test Machines: Eight

fast1, fast2 a Linux 2.4.20 machine with FAST TCP kernel patches, at SoX

fast3 a Linux 2.4.20 machine with conventional TCP, at SoX

fast4 a Linux 2.4.20 machine with FAST TCP kernel patches, at NC-ITEC

fast5 a Linux 2.4.20 machine with FAST TCP kernel patches, at PNW

fast6 a Linux 2.4.20 machine with conventional TCP, at PSC

gigatcp1 a FreeBSD 4.3-RELEASE machine with conventional TCP, at SoX

nms1-sttl a FreeBSD 4.6-RELEASE machine with conventional TCP, at STTL

Test Paths: Four

path1 nms1-sttl $\rightarrow$ gigatcp1 (stock FreeBSD Reno),

path2 fast4 $\rightarrow$ fast1 (Linux with FAST patches),

path3 fast5 $\rightarrow$ fast2 (Linux with FAST patches),

path4 fast6 $\rightarrow$ fast3 (stock Linux Reno).

- Individual bottlenecks: 1Gb/s at sender NIC (and before receiver)
- Collective bottleneck: 2.5Gb/s at OC-48 from ATLA to SoX

Test Methodology

- One TCP connection on each path
- Staggered start (one connection per 15 minutes)
- Unlimited supply of data
  - Memory-to-memory copy
  - Three connections congestion-limited
  - One connection limited by CPU speed on sender—to measure loss and delay for cross-traffic
- Throughput and delay measurements on all paths

Measuring Delay

- FAST TCP patches log kernel messages with various parameters of internal state
- For non-FAST connections, a custom tool was used to measure delay
- Same tool also used for FAST connections, so these come with two sets of data

Custom Throughput and Delay Measurement Tool

- Client/server architecture
- Client requests window size and I/O block size
- Client sends blocks of data with timestamps in the beginning
- Server echoes timestamps
- Client prints machine-readable statistics
- Source code available at http://www.internet2.edu/~shalunov/i2perf/

Baseline Tests

- Conducted on each path sequentially
- No temporal overlap
- path1 is limited by CPU speed
- The rest get good performance
- Plots are in the paper but not this talk

Concurrent Test Schedule

Test conducted on Nov 11, 2003 between 2 am and 6 am ET

Second	Event
0	path1 starts
900	path2 starts
1800	path3 starts
2700	path4 starts
3600	
4500	path2 finishes
5400	path3 finishes
6300	path4 finishes
7200	path1 finishes

Overall test results

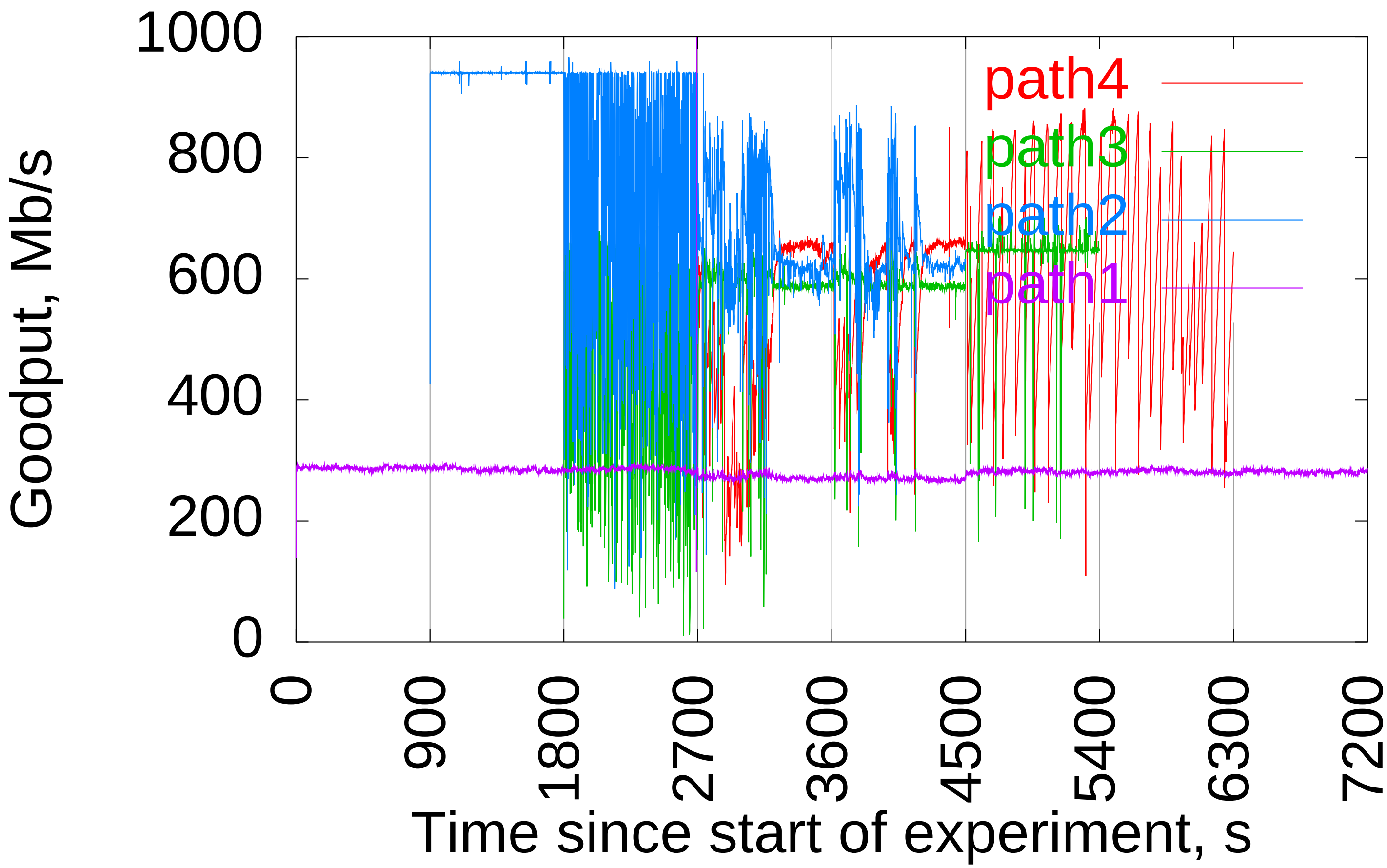
Baseline

Path	Mb/s	min rtt	avg rtt	max rtt
path1	286.100	58.751	61.419	1868.287
path2	938.460	44.250	62.850	802.687
path3	650.589	59.378	100.497	1600.684
path4	602.295	46.129	91.357	851.856

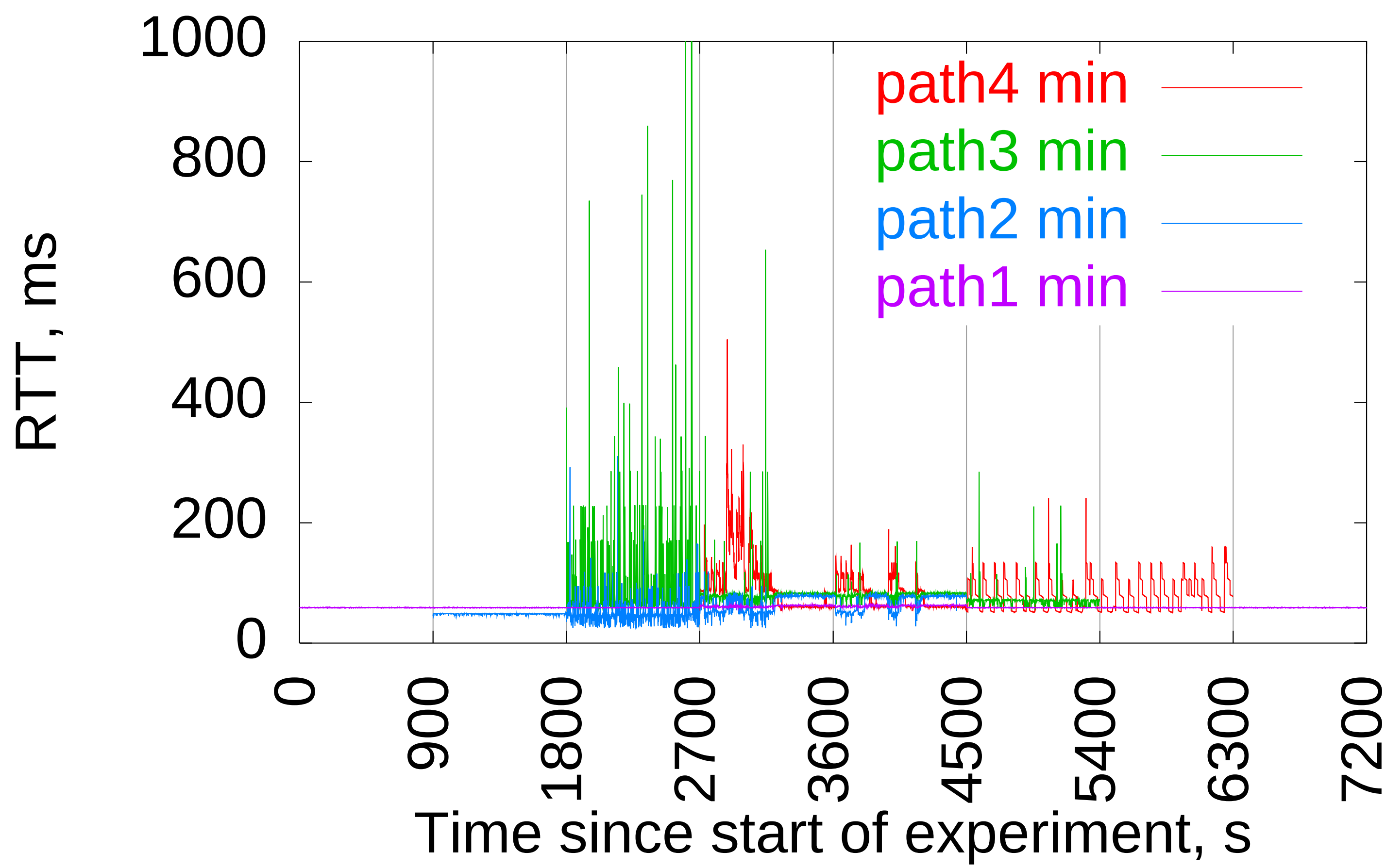
Concurrent

Path	Mb/s	min rtt	avg rtt	max rtt
path1	280.411	58.643	61.143	1111.416
path2	747.636	25.068	77.155	1090.875
path3	574.248	58.788	111.707	3604.343
path4	577.297	50.610	96.492	957.372

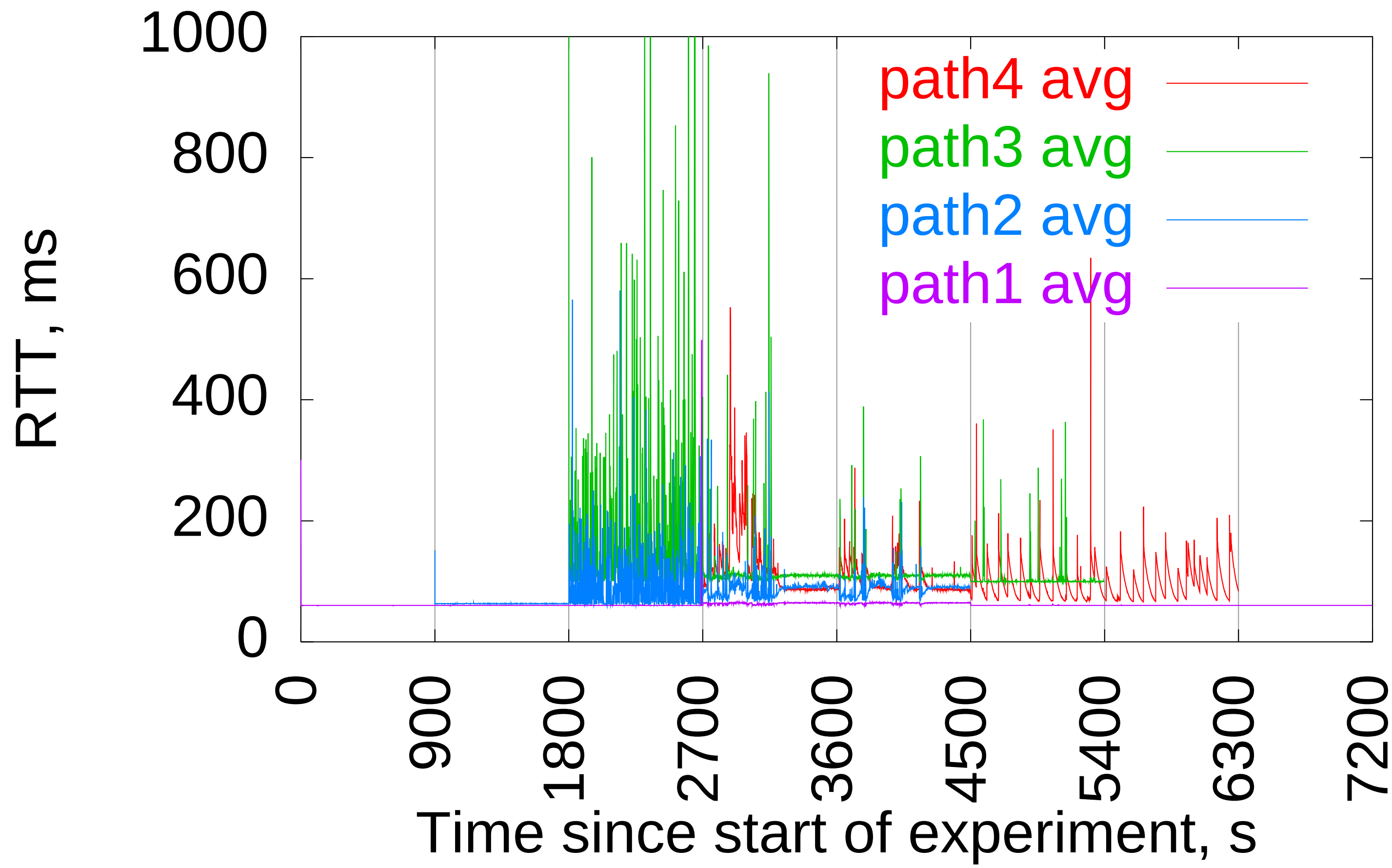
Throughput of all paths (concurrent experiment)



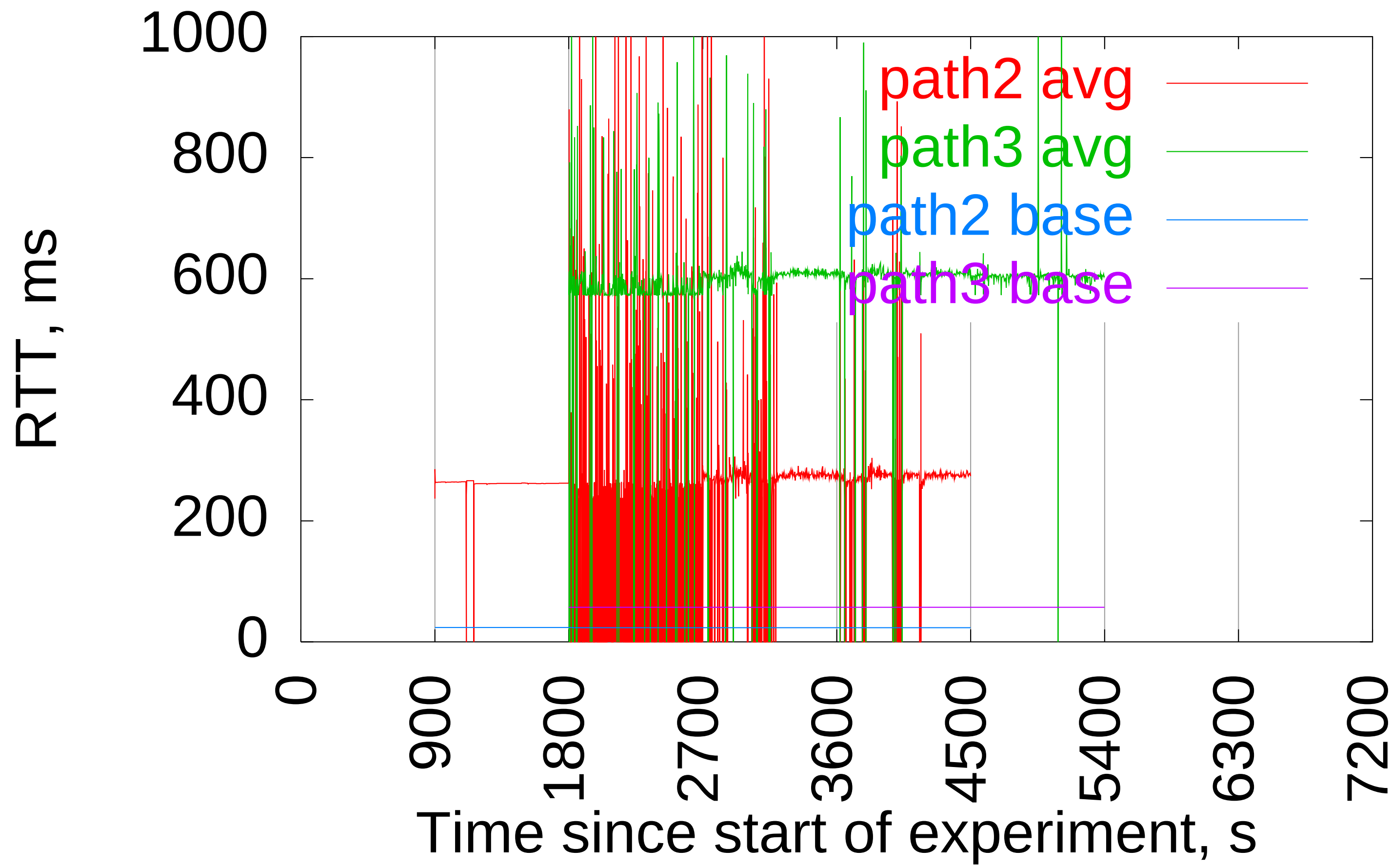
Minimum round-trip time of all paths (concurrent experiment, user space)



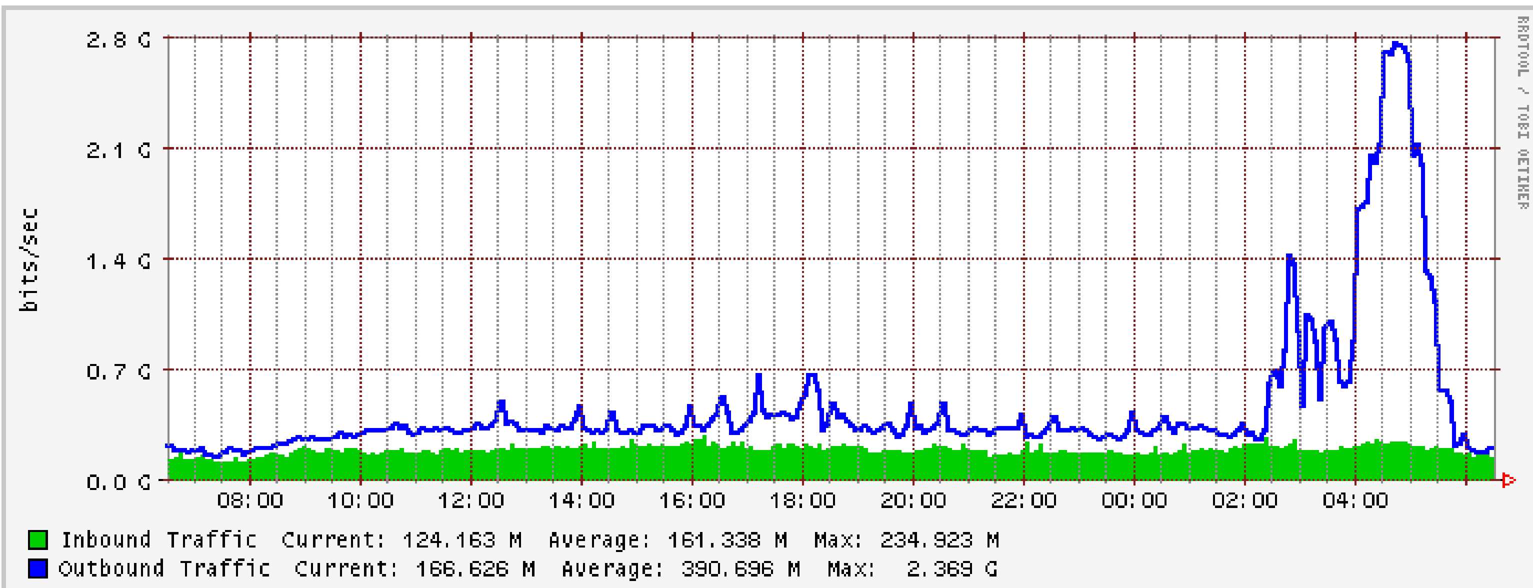
Average round-trip time of all paths (concurrent experiment, user space)



Round-trip time of path2 and path3 (concurrent experiment, kernel space)



Bottleneck link utilization (from SNMP data)



Observations

- Bottleneck link is congested
- Bottleneck link never loses even a single packet belonging to the path1 flow
- path1 flow packets are not significantly delayed at bottleneck
- Internal queueing on all Linux paths
- More stable throughput on Linux at time of congestion

Conclusions

- FAST TCP appears to allow bulk transfers that saturate the link without affecting conventional traffic
  - No QoS
  - No AQM
- Linux (with or without FAST TCP patches) appears to build up internal queues that affect TCP congestion control in a profound manner

Acknowledgments

The tests used FAST TCP machines owned and operated by Caltech, with the help of Steven Low, Cheng Jin, and Raj Jayaraman. Cas D'Angelo (Southern Crossroads GigaPoP), David Richardson (Pacific Northwest GigaPoP), Matt Mathis (Pittsburgh Supercomputing Center), and John Moore (North Carolina Internet2 Technology Evaluation Center) provided high-speed hosting for the test machines.

Contact

Stanislav Shalunov ⟨shalunov@internet2.edu⟩

Source code

http://www.internet2.edu/~shalunov/i2perf/