A Systematic Analysis of TCP Performance

Yee-Ting Li, Steven Dallison,
Richard Hughes-Jones and Peter Clarke

University College London &
Manchester University

Motivation

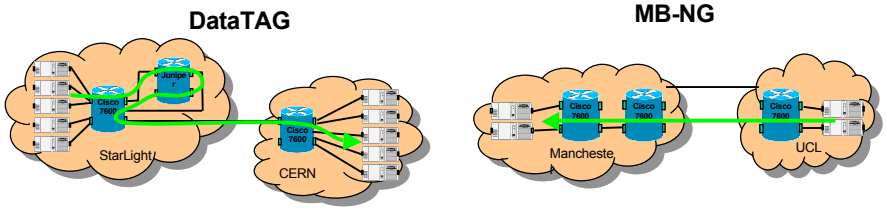
- TCP does not perform very well under certain environments
- New TCP stacks being proposed
- How to take advantage of capacity?
- Are TCP stacks sufficient for high speed transport?
- More importantly; is it sufficient for high speed data replication/movement?
- What are the bottlenecks?

Overview

- TCP analysis
  - How does New TCP perform under real simulated environments?
  - Quantify effects on background traffic
  - How do these protocols scale?
- RAID tests
  - How quickly can we get real data on/off disks?
  - Kernel parameters
- Transfer Programs
  - What happens when we try to move real data?

Introduction

- TCP stacks
  - Scalable TCP, HSTCP, H-TCP
- Networks



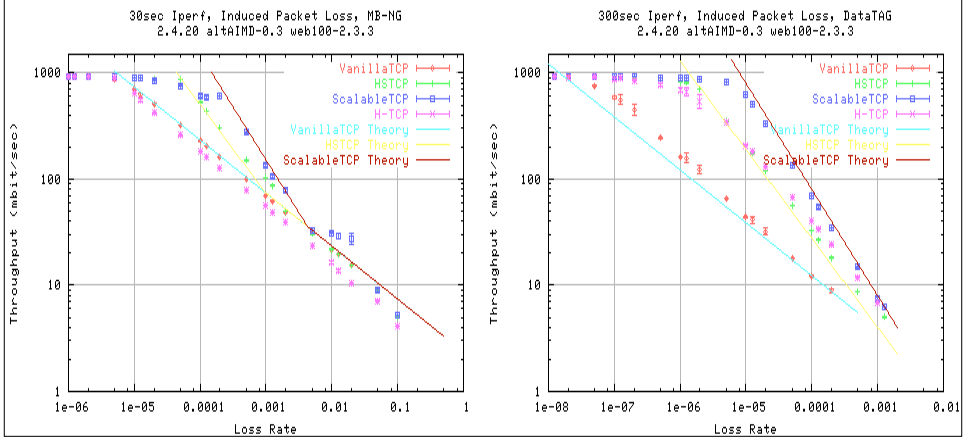
Bottleneck Capacity	1Gb/sec	Bottleneck Capacity	1Gb/sec
RTT	120msec	RTT	6msec

altAIMD Linux Kernel

- Modified 2.4.20 kernel
  - SACK Patch
  - On-the-fly switchable between HSTCP, Scalable, GridDT and H-TCP
  - ABC (RFC3465)
  - Web100 (2.3.3)
  - Various switches to turn parts of TCP on/off
- Large TXQueueLens
- Large netdev_max_backlog

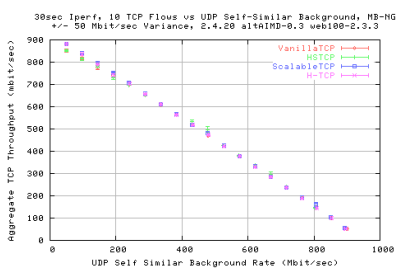
Response Function

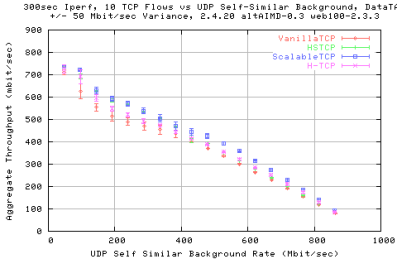
- Induced packet drop at receiver (kernel modification)



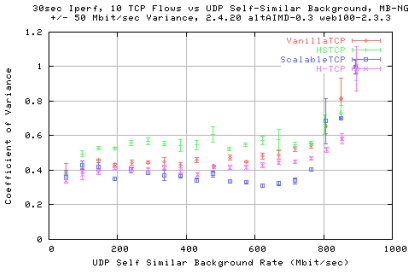
10 TCP Flows versus Self-Similar Background

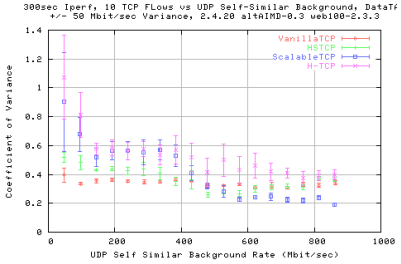
Aggregate BW





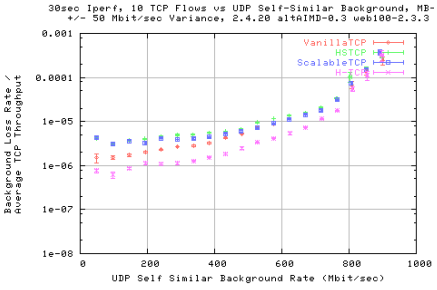
CoV

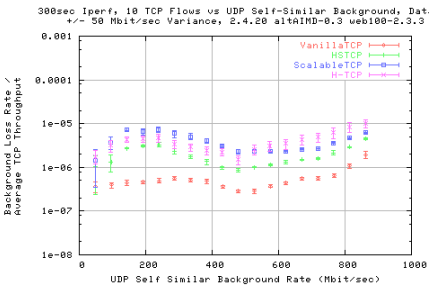




10 TCP Flows versus Self-Similar Background

BG Loss per TCP BW

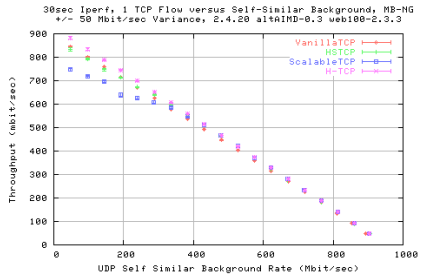


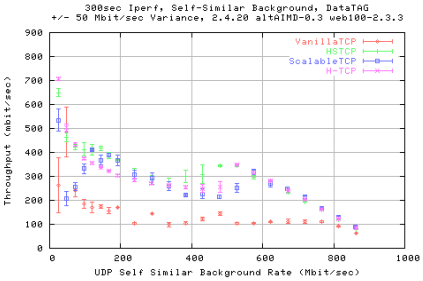


Single TCP Flow versus Self-Similar Background

- Deviation from expected performance
- Not because of protocol…

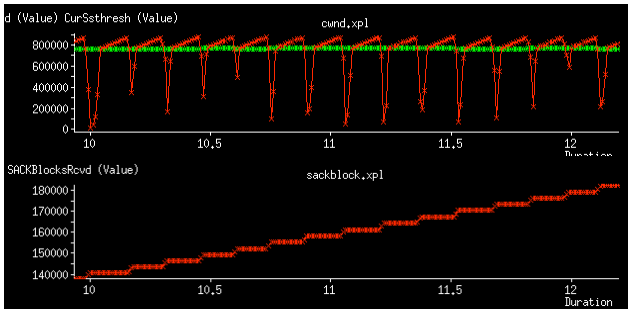
1 TCP Flow





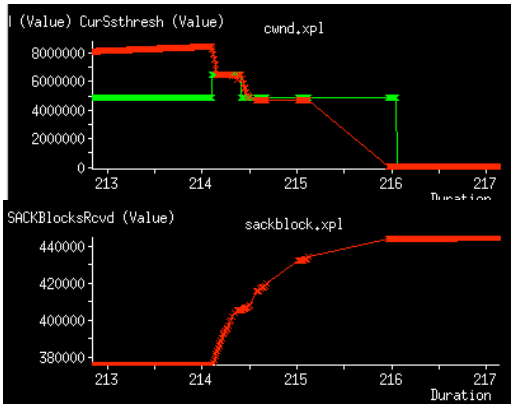
SACKs

- Implementation problems in Linux
  → Use Tom's SACK fast-path patch
- Still not sufficient:



Scalable TCP on MB-NG with 200mbit/sec CBR Background

SACK Processing overhead



Periods of web100 silence due to high cpu utilization?
Logging done in userspace – kernel time taken up by tcp sack processing?
Why is cwnd set to low values after?

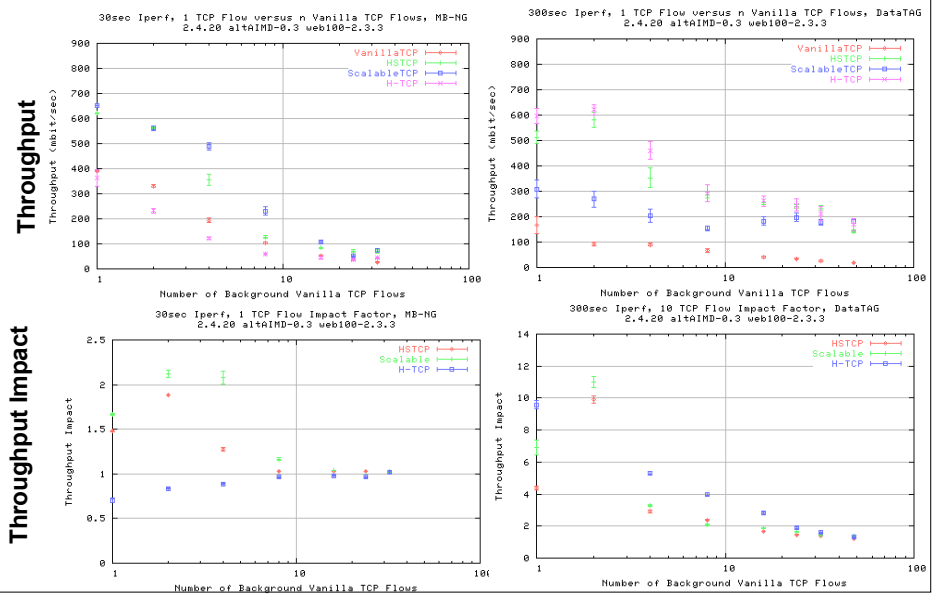
Impact

- New stacks are designed to get high throughput
  - Achieved by penalising throughput of other flows
  - Naturally 'unfair' – but it's the inherent design of these protocols
  - Describe through the effect on background traffic.
- Impact

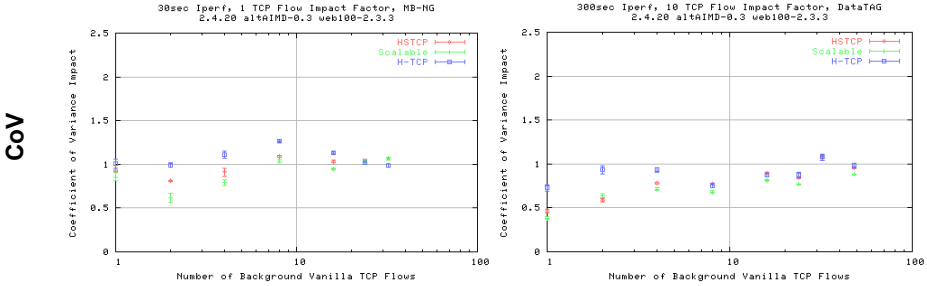
$$\text{BW impact} = \frac{\text{throughput of n-Vanilla flows}}{\text{throughput of (n-1) Vanilla flows + 1 new TCP flow}}$$

- Describes ratio of achieved metric with and without new TCP flow(s)

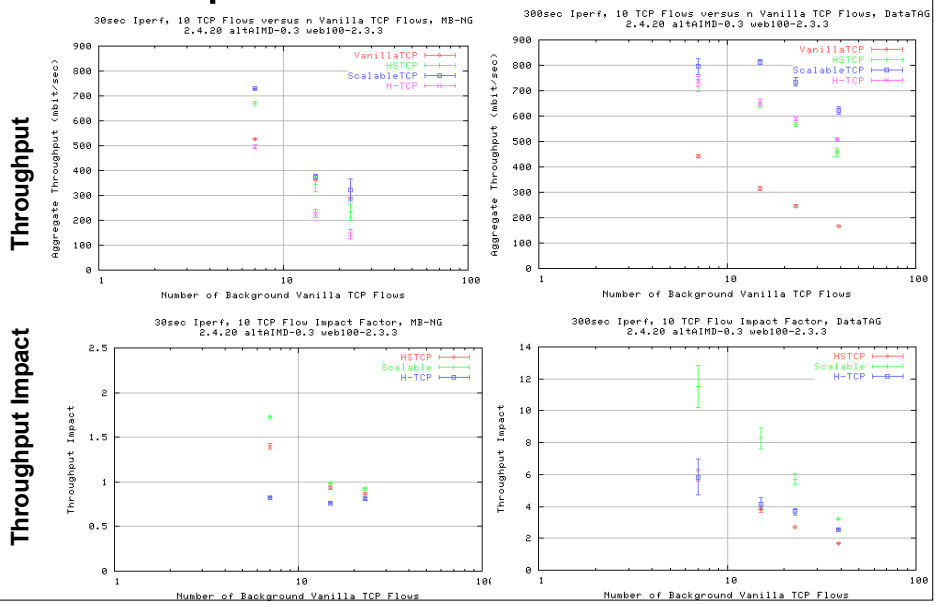
Impact of 1 TCP Flow



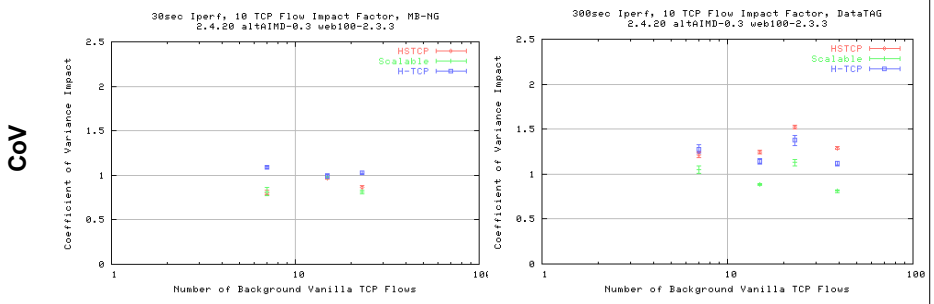
1 New TCP Impact



Impact of 10 TCP Flows

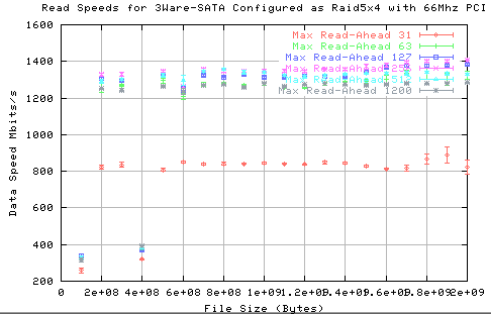


10 TCP Flows Impact



RAID Performance

- Test of RAID cards
  - 33Mhz & 66Mhz
  - RAID0 (striped) & RAID5 (stripped with redundancy)
  - Kernel parameters
- Tested on Dual 2.0Ghz Xeon Supermicro P4DP8-G2 motherboard
- Disk; Maxstor 160GB 7200rpm 8MB



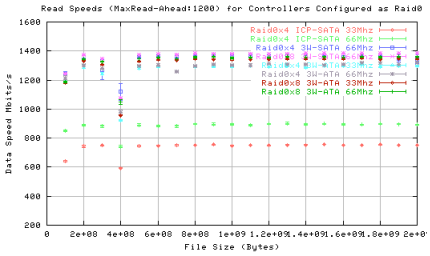
Read_Ahead kernel tuning

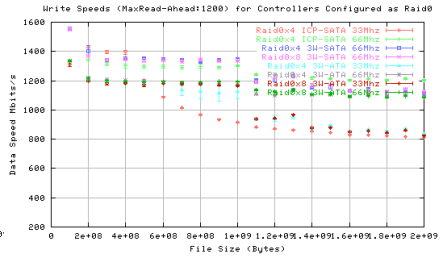
/proc/sys/vm/max-readahead

RAID Controller Performance

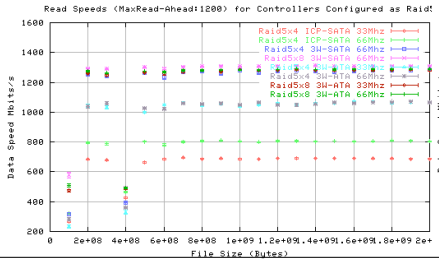
Read Speed | **Write Speed**

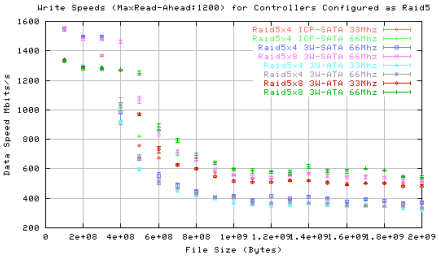
RAID 0





RAID 5





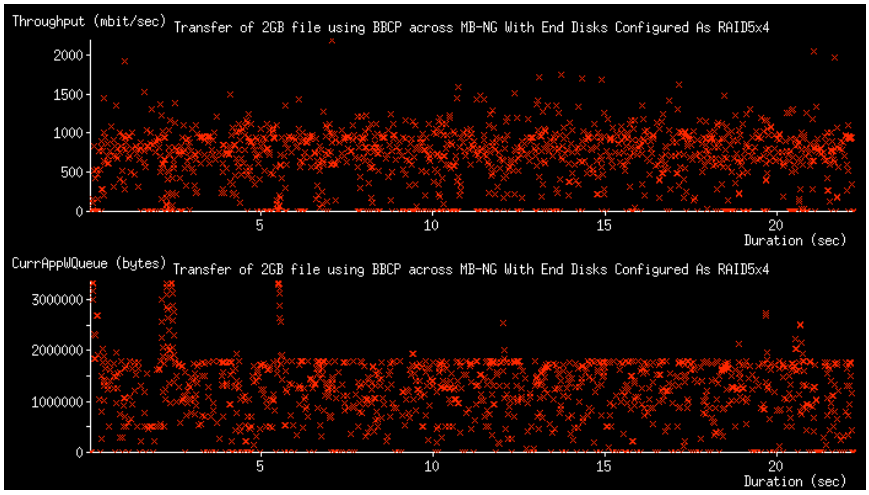
RAID Summary

Controller Type	Number of Disks	Read Speed Raid 0 (Mbits/s)	Write Speed Raid 0 (Mbits/s)	Read Speed Raid 5 (Mbits/s)	Write Speed Raid 5 (Mbits/s)
ICP 33	4	**751**	**811**	**686**	490
ICP 66	4	893	**1202**	804	538
3W-ATA 33	4	1299	835	1065	**319**
3W-ATA 66	4	1320	1092	1066	335
3W-ATA 33	8	1344	824	1280	482
3W-ATA 66	8	1359	1085	1289	**541**
3W-SATA66	4	1343	1116	1283	372
3W-SATA 66	8	**1389**	1118	**1310**	513

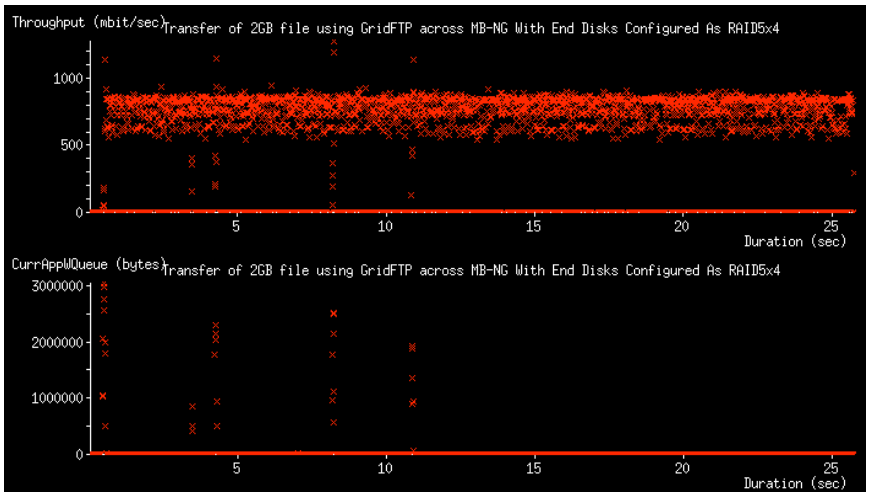
Replication Programs

- Transfer on MBNG
  - 3WARE source
  - ICP sink
  - RAID5 o RAID5
  - Limited to ∼800mbit/sec
  - Single flow
- Bottleneck is socket buffer
  - AIMD independent
- BBCP & GridFTP

bbcp



GridFTP



Summary

- TCP stack performances
  - Major issues with running at high throughput due to Linux implementations
- RAID
  - RAID5 more useful
  - ICP good for writing, 3WARE better for reading
- Program problems