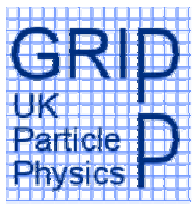
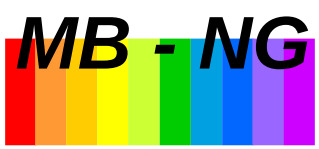
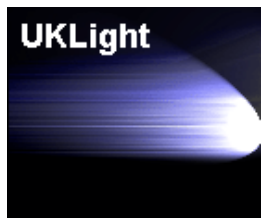
End-user systems: NICs, MotherBoards, Disks, TCP Stacks & Applications

Richard Hughes-Jones

Work reported is from many Networking Collaborations







Network Performance Issues

End System Issues

- Network Interface Card and Driver and their configuration
- Processor speed
- MotherBoard configuration, Bus speed and capability
- Disk System
- TCP and its configuration
- Operating System and its configuration

Network Infrastructure Issues

- Obsolete network equipment
- Configured bandwidth restrictions
- Topology
- Security restrictions (e.g., firewalls)
- Sub-optimal routing
- Transport Protocols

Network Capacity and the influence of Others!

- Congestion – Group, Campus, Access links
- Many, many TCP connections
- Mice and Elephants on the path

Methodology used in testing NICs & Motherboards

Latency Measurements

- **UDP/IP packets sent between back-to-back systems**
  - Processed in a similar manner to TCP/IP
  - Not subject to flow control & congestion avoidance algorithms
  - Used UDPmon test program

- **Latency**
- Round trip times measured using Request-Response UDP frames
- Latency as a function of frame size
  - Slope is given by:

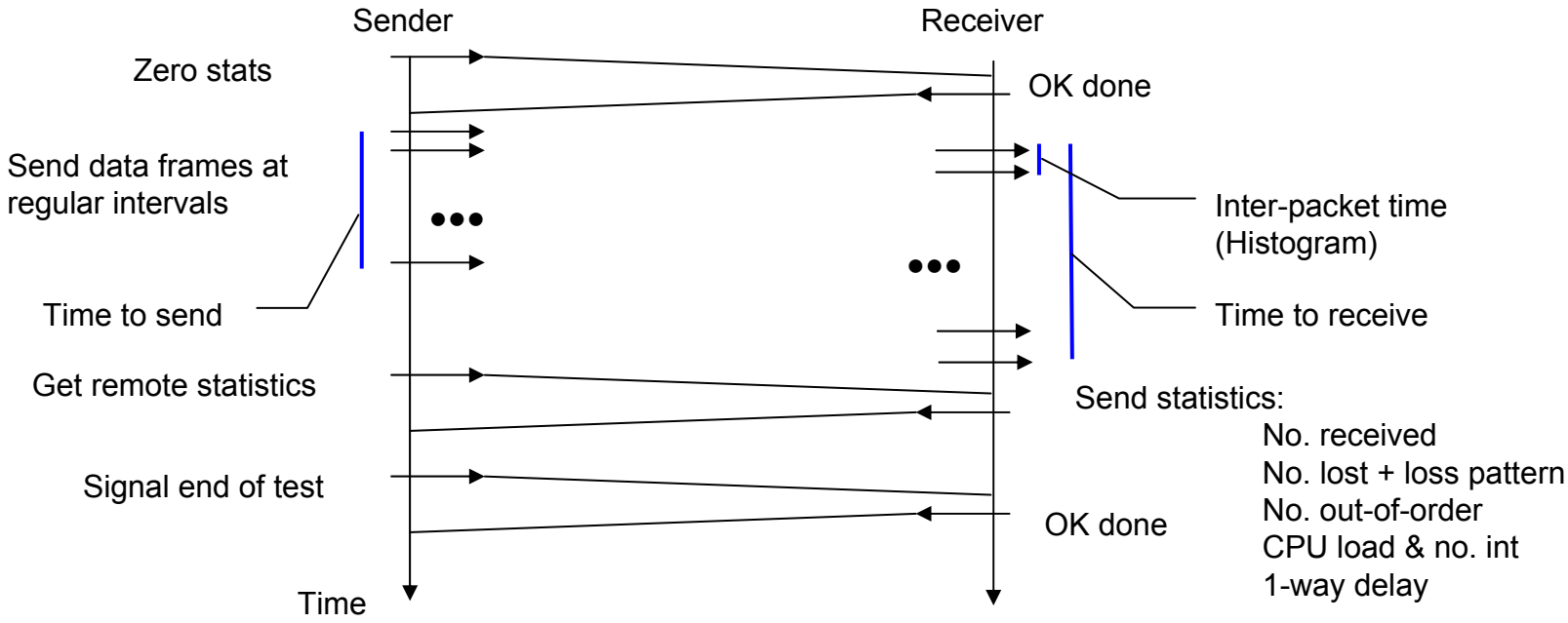
$$s = \sum_{\text{data paths}} \frac{1}{\frac{db}{dt}}$$

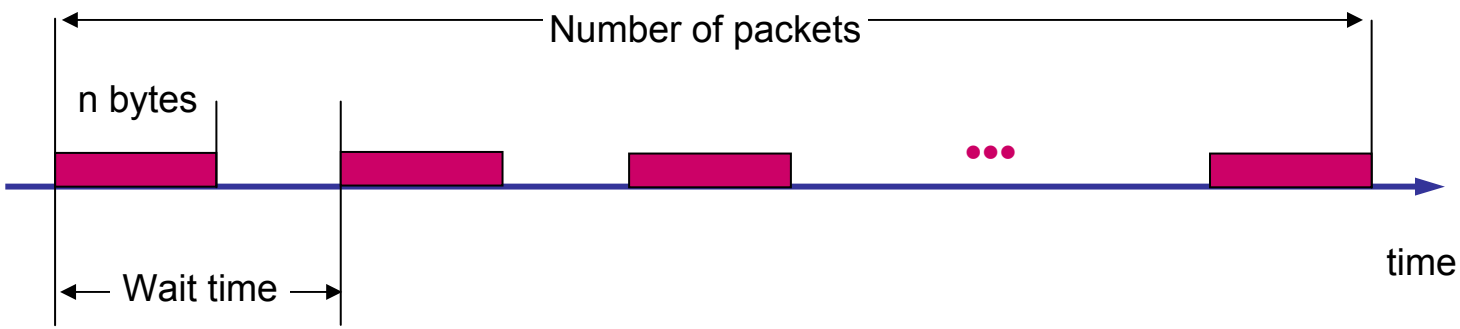
    - Mem-mem copy(s) + pci + Gig Ethernet + pci + mem-mem copy(s)

  - Intercept indicates: processing times + HW latencies
- Histograms of ‘singleton’ measurements
- Tells us about:
  - Behavior of the IP stack
  - The way the HW operates
  - Interrupt coalescence

Throughput Measurements

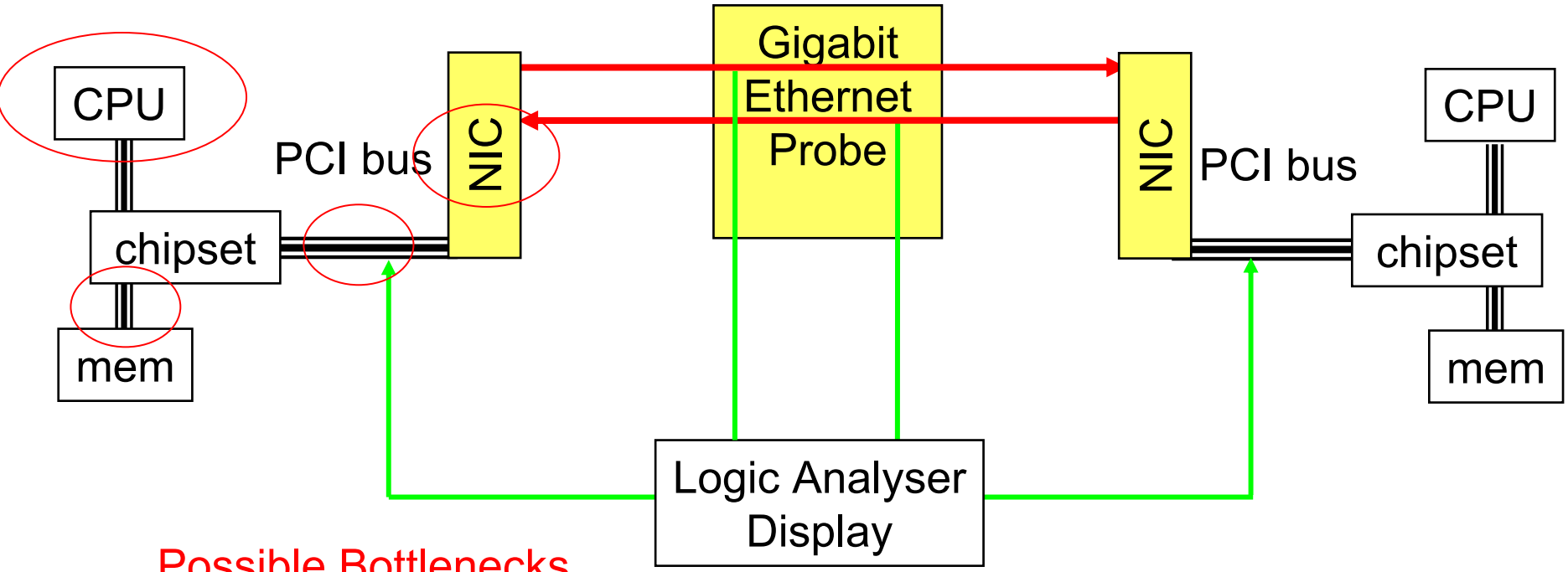
- **UDP Throughput**
- Send a controlled stream of UDP frames spaced at regular intervals





PCI Bus & Gigabit Ethernet Activity

- **PCI Activity**
- Logic Analyzer with
  - PCI Probe cards in sending PC
  - Gigabit Ethernet Fiber Probe Card
  - PCI Probe cards in receiving PC

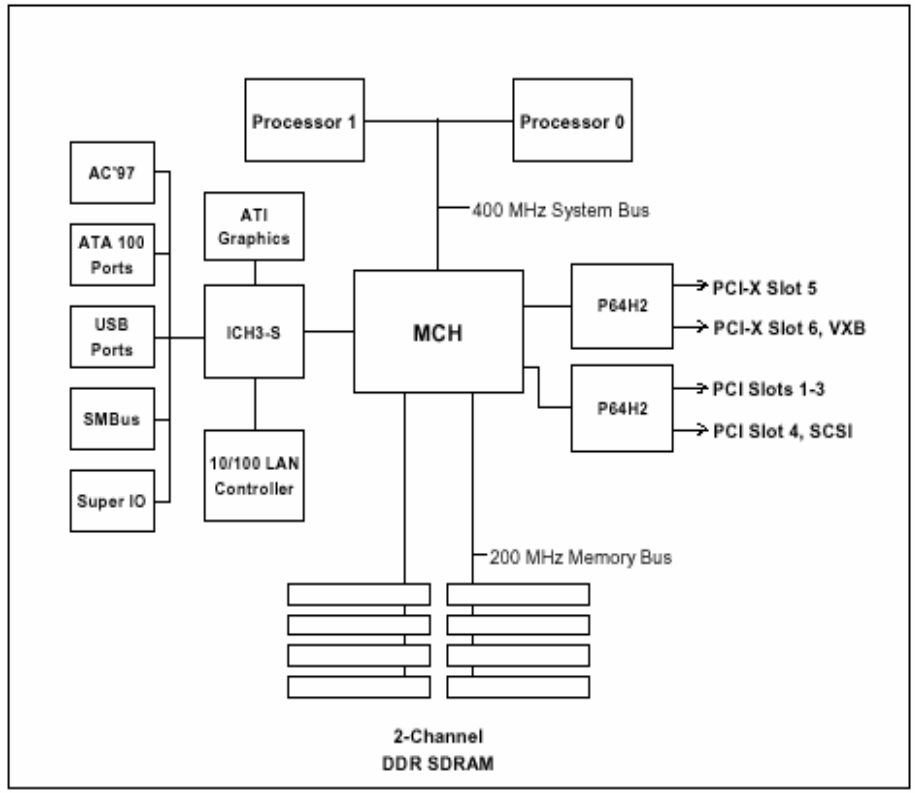


"Server Quality" Motherboards

- **SuperMicro P4DP8-2G (P4DP6)**
- **Dual** Xeon
- 400/522 MHz Front side bus

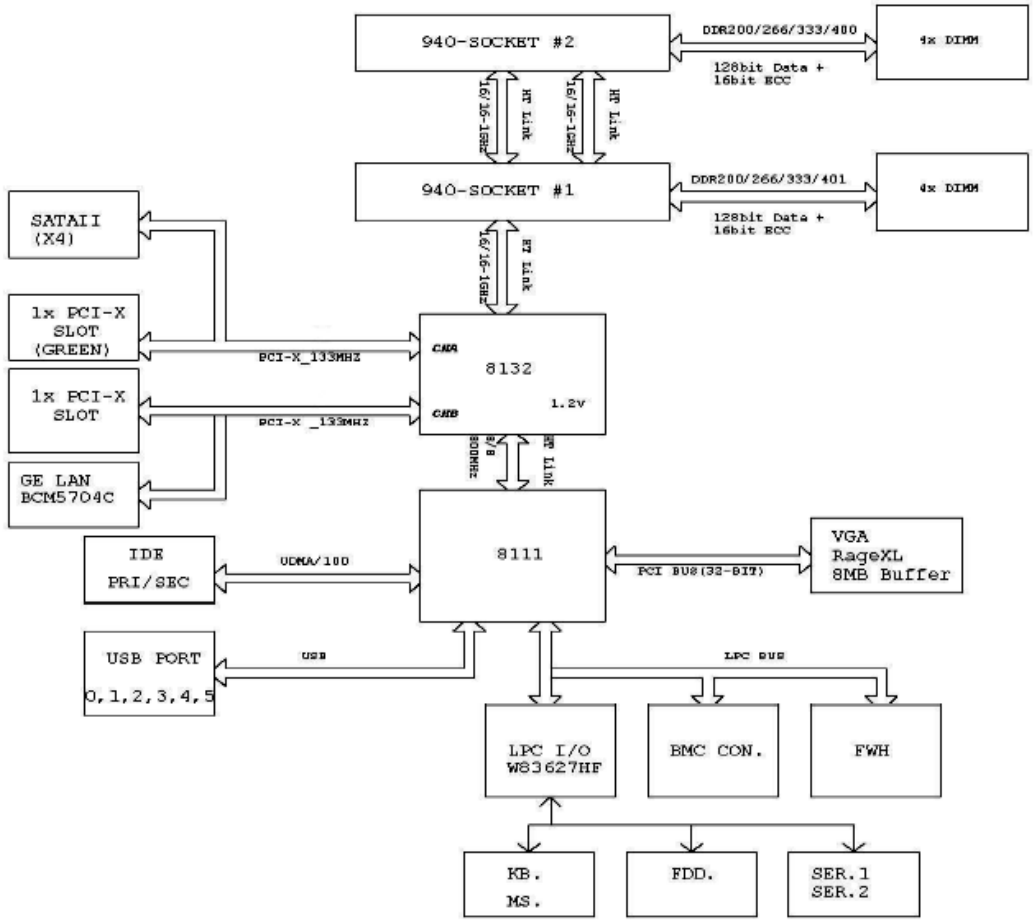
- **6 PCI PCI-X slots**
- **4 independent PCI buses**
  - 64 bit 66 MHz PCI
  - 100 MHz PCI-X
  - 133 MHz PCI-X

- Dual Gigabit Ethernet
- Adaptec AIC-7899W dual channel SCSI
- UDMA/100 bus master/EIDE channels
  - data transfer rates of 100 MB/sec burst



"Server Quality" Motherboards

- **Boston/Supermicro H8DAF**
- **Two Dual Core Opterons**
- 200 MHz DDR Memory
  - Theory BW: 6.4Gbit
- **HyperTransport**
- **2 independent PCI buses**
  - 133 MHz PCI-X
- 2 Gigabit Ethernet
- SATA
- (PCI-e)



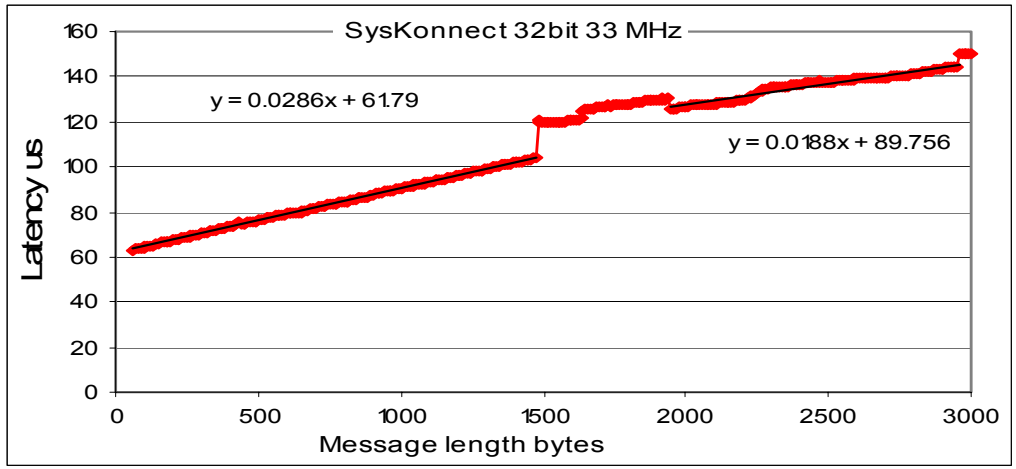
BRUNEL UNIVERSITY WEST LONDON

NIC & Motherboard Evaluations

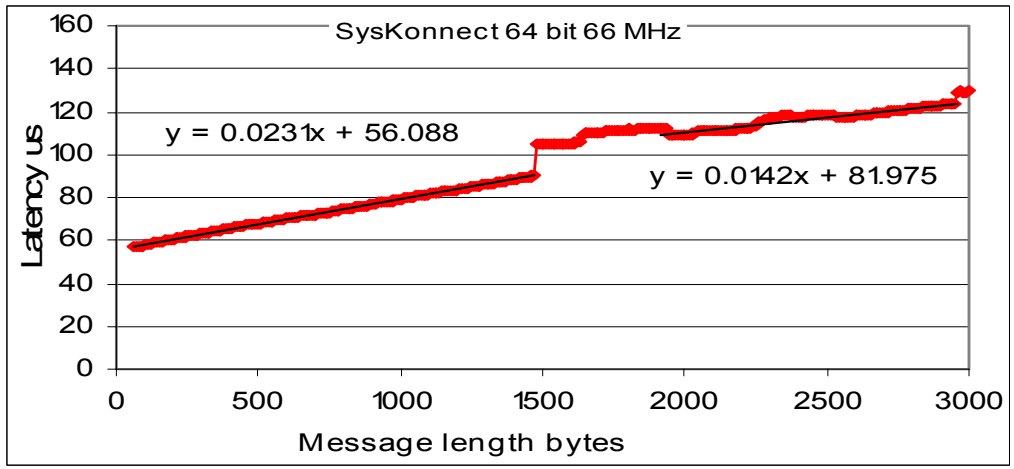
SuperMicro 370DLE: Latency: SysKonnect

- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz
- RedHat 7.1 Kernel 2.4.14

- **PCI:32 bit 33 MHz**
- Latency small 62 µs & well behaved
- Latency Slope **0.0286 µs/byte**
- Expect: **0.0232 µs/byte**
  - PCI 0.00758
  - GigE 0.008
  - PCI 0.00758

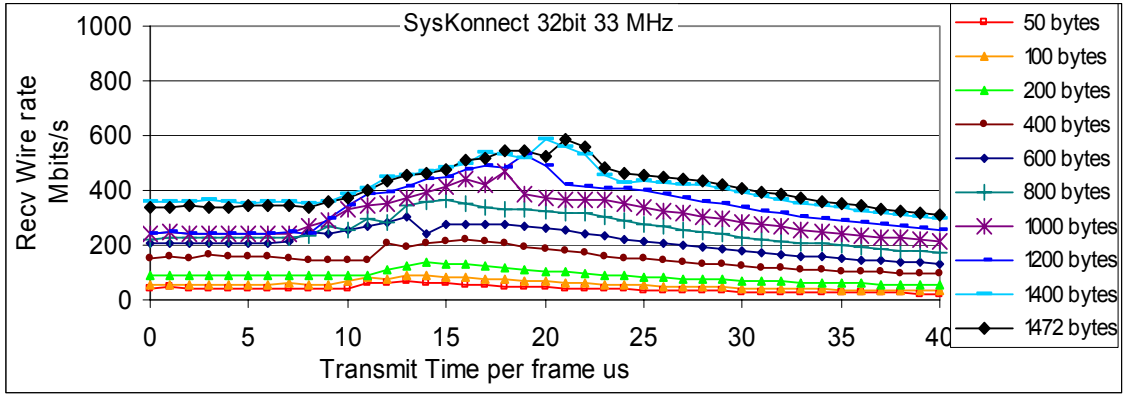


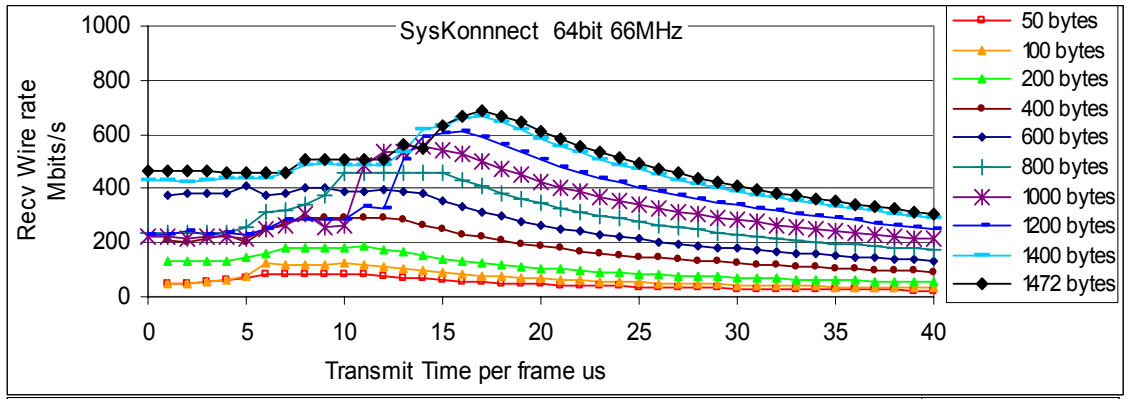
- **PCI:64 bit 66 MHz**
- Latency small 56 µs & well behaved
- Latency Slope **0.0231 µs/byte**
- Expect: **0.0118 µs/byte**
  - PCI 0.00188
  - GigE 0.008
  - PCI 0.00188
- Possible extra data moves ?

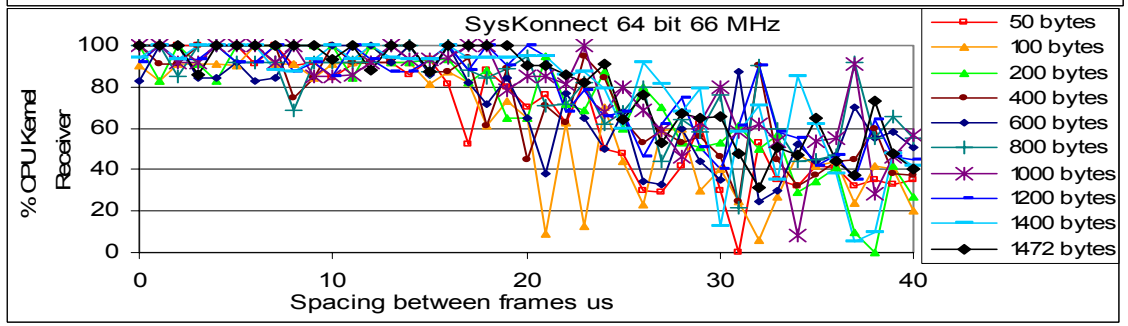


SuperMicro 370DLE: Throughput: SysKonnect

- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz
- RedHat 7.1 Kernel 2.4.14
- **PCI:32 bit 33 MHz**
- Max throughput **584Mbit/s**
- No packet loss >18 us spacing
- **PCI:64 bit 66 MHz**
- Max throughput **720 Mbit/s**
- No packet loss >17 us spacing
- Packet loss during BW drop
- **95-100% Kernel mode**

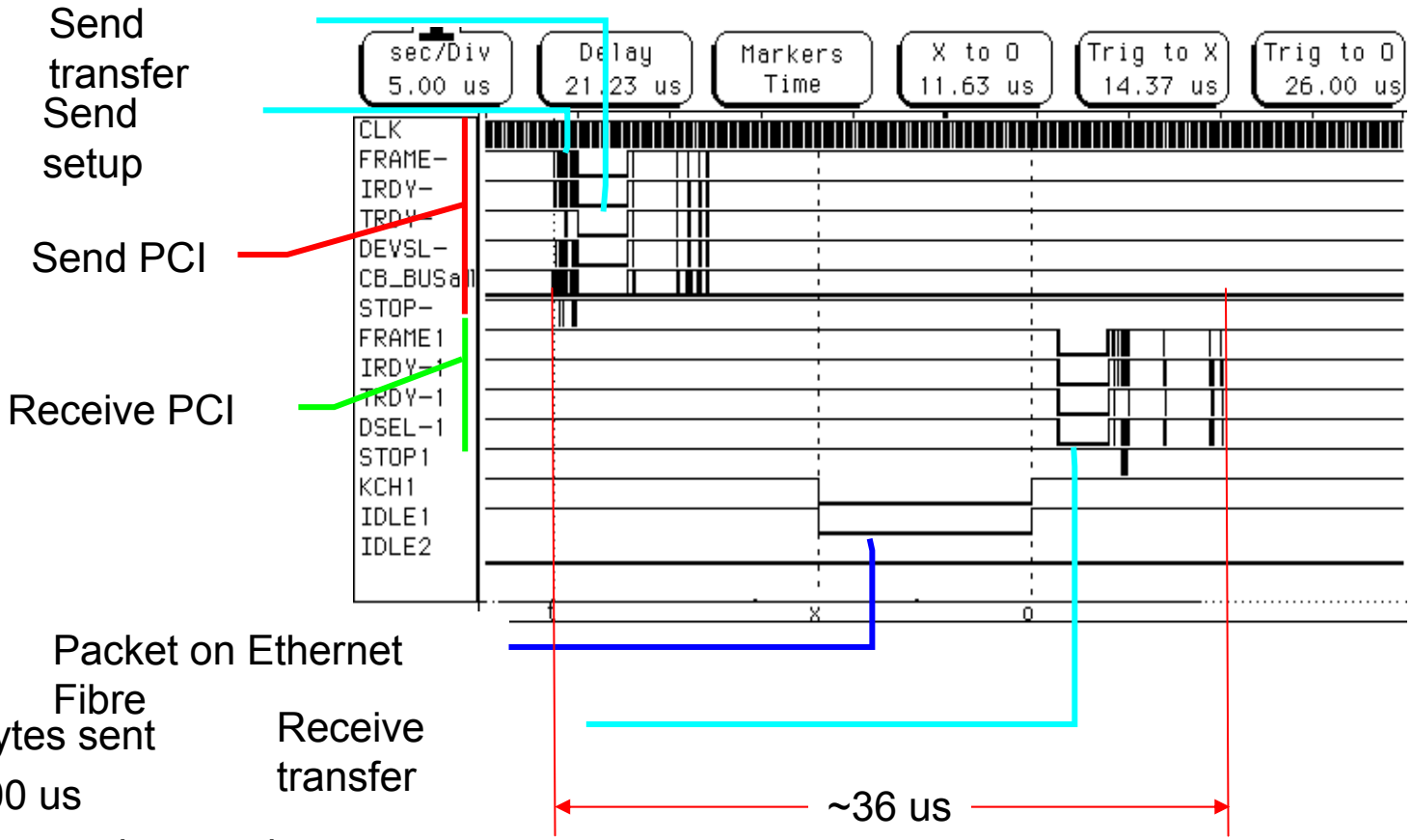






SuperMicro 370DLE: PCI: SysKonnect

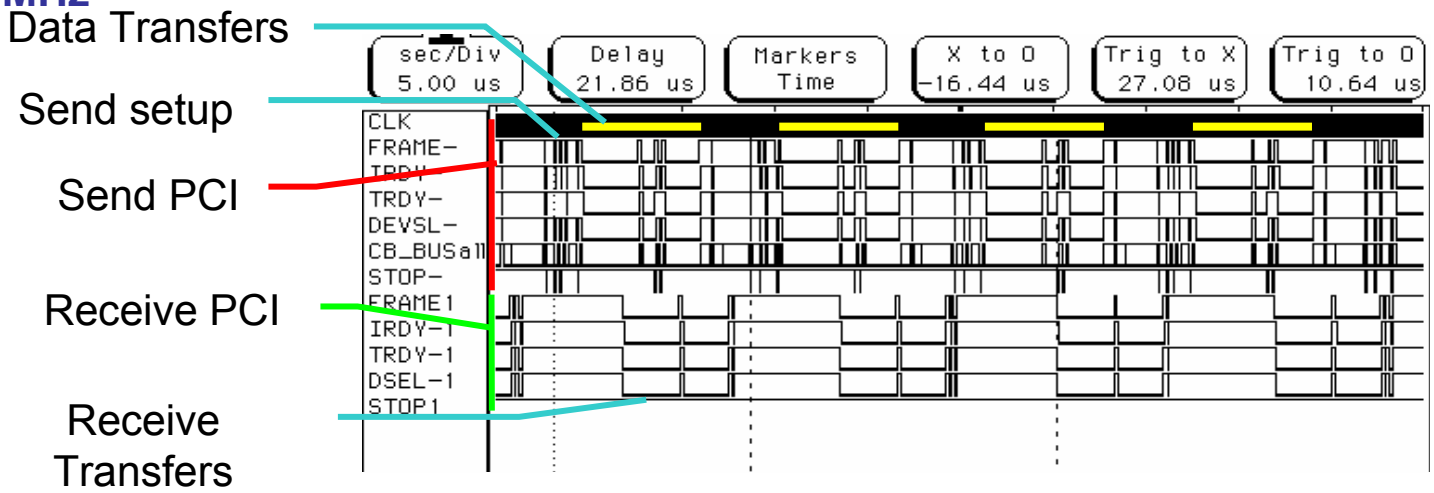
- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz **PCI:64 bit 66 MHz**
- RedHat 7.1 Kernel 2.4.14



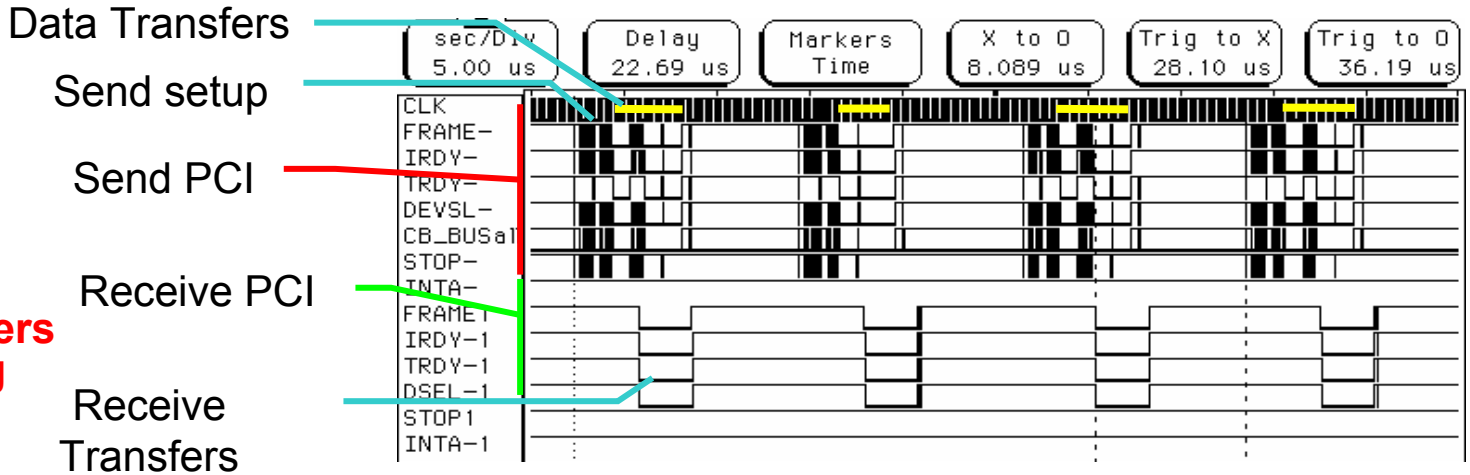
- 1400 bytes sent
- Wait 100 us
- ~8 us for send or receive
- Stack & Application overhead ~ 10 us / node

Signals on the PCI bus

- 1472 byte packets every 15 µs Intel Pro/1000
- PCI:64 bit 33 MHz
- 82% usage

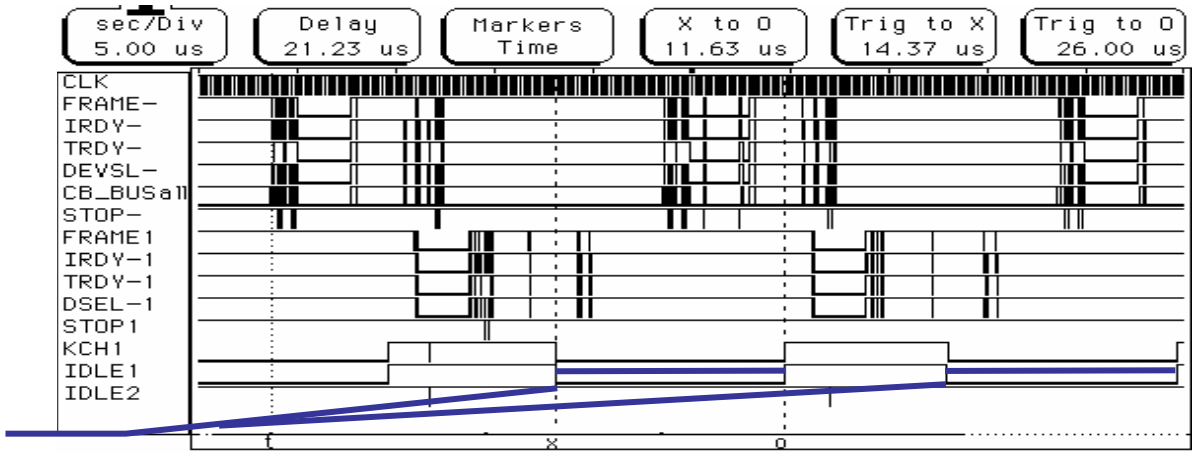


- PCI:64 bit 66 MHz
- 65% usage
- Data transfers half as long



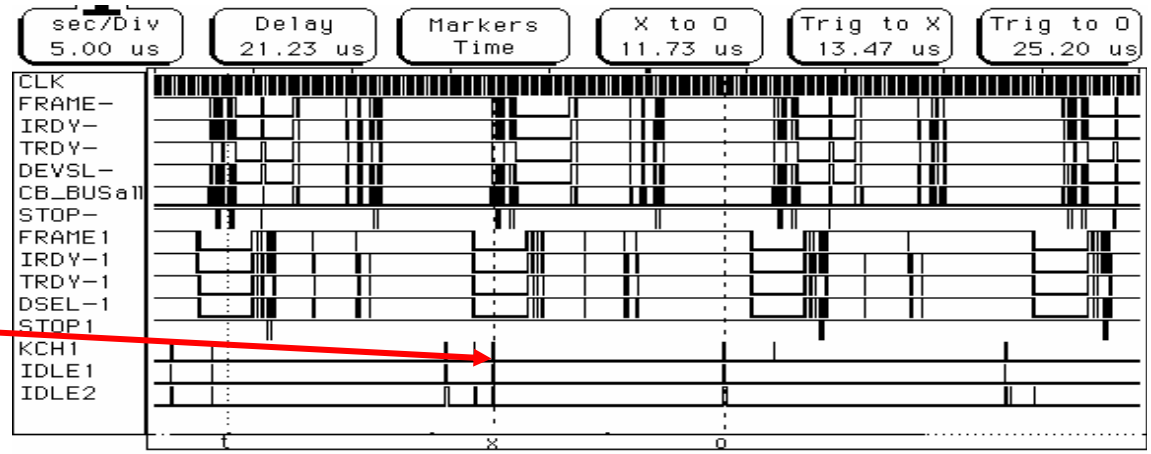
SuperMicro 370DLE: PCI: SysKonnect

- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz **PCI:64 bit 66 MHz**
- RedHat 7.1 Kernel 2.4.14
  - 1400 bytes sent
  - Wait 20 us



Frames on Ethernet Fiber 20 us spacing

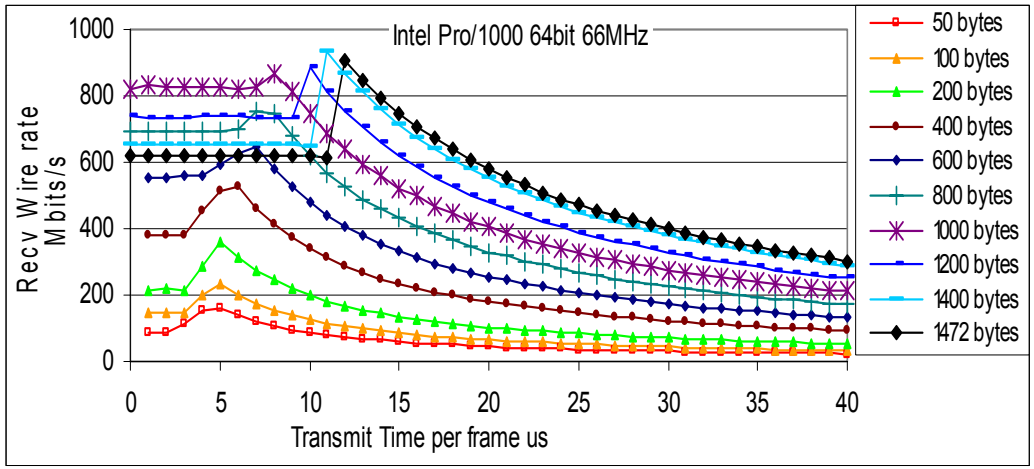
- 1400 bytes sent
- Wait 10 us



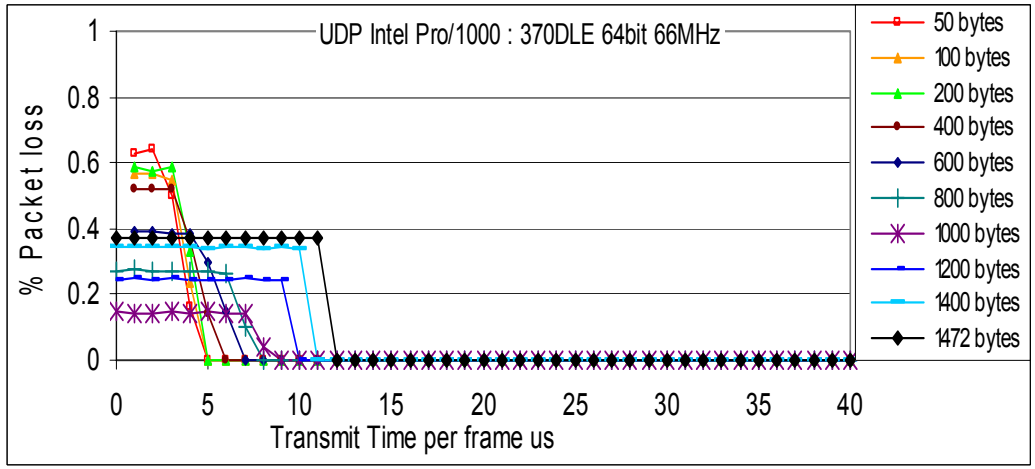
Frames are back-to-back
800 MHz Can drive at line speed
Cannot go any faster !

SuperMicro 370DLE: Throughput: Intel Pro/1000

- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz **PCI:64 bit 66 MHz**
- RedHat 7.1 Kernel 2.4.14
- Max throughput **910 Mbit/s**
- No packet loss >12 us spacing

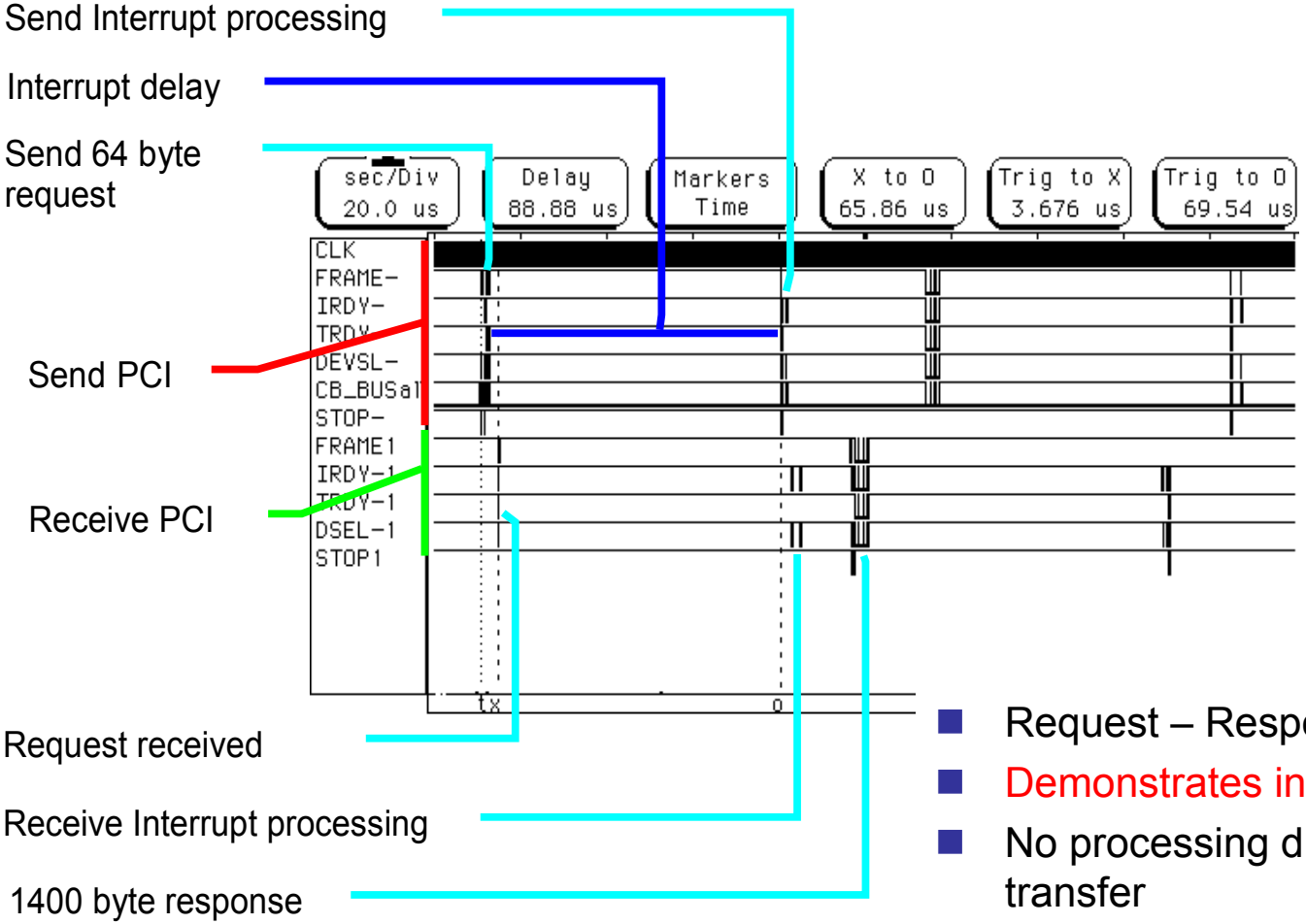


- Packet loss during BW drop
- CPU load 65-90% spacing < 13 us



SuperMicro 370DLE: PCI: Intel Pro/1000

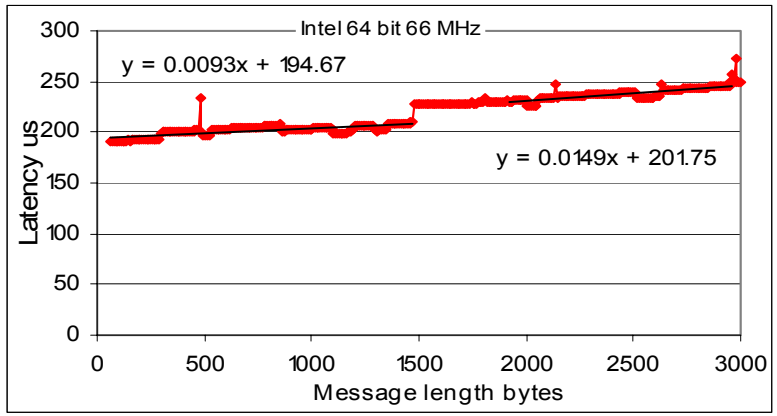
- Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset
- CPU: PIII 800 MHz **PCI:64 bit 66 MHz**
- RedHat 7.1 Kernel 2.4.14



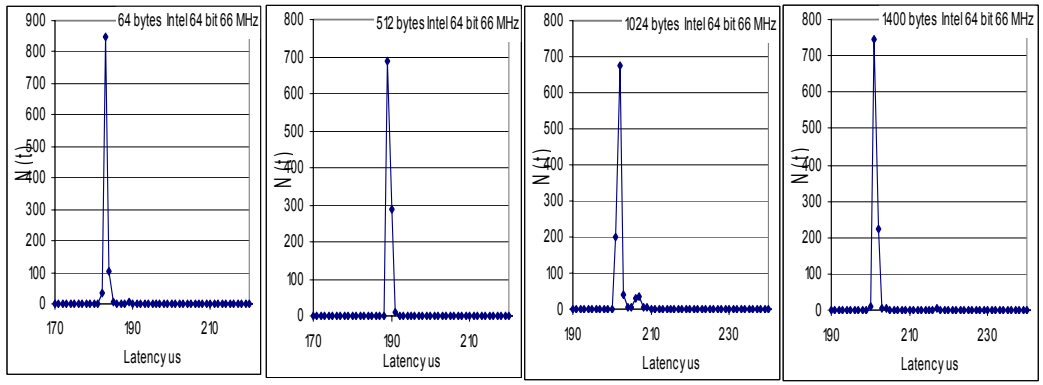
- Request – Response
- Demonstrates interrupt coalescence
- No processing directly after each transfer

SuperMicro P4DP6: Latency Intel Pro/1000

- Motherboard: SuperMicro P4DP6 Chipset: Intel E7500 (Plumas)
- CPU: Dual Xeon Prestonia 2.2 GHz **PCI, 64 bit, 66 MHz**
- RedHat 7.2 Kernel 2.4.19



- Some steps
- Slope **0.009 us/byte**
- Slope flat sections : **0.0146 us/byte**
- Expect 0.0118 us/byte

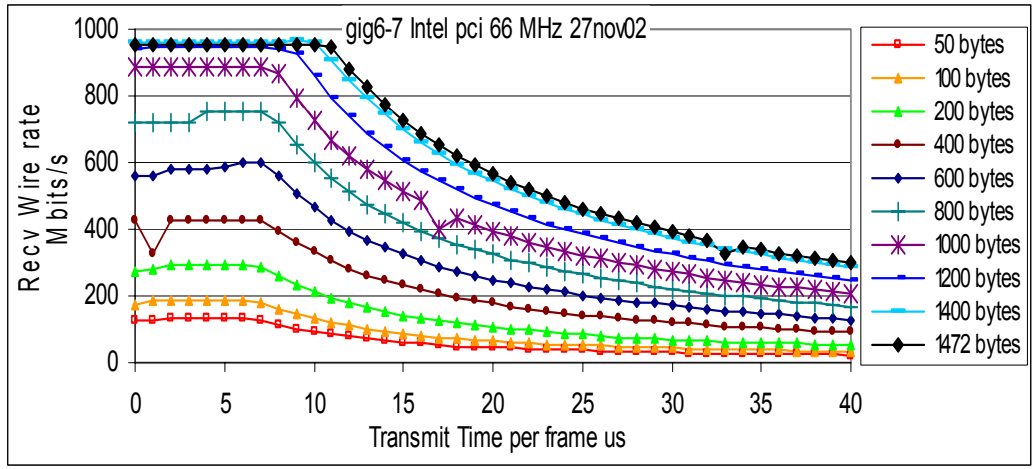


- No variation with packet size
- FWHM 1.5 us
- Confirms timing reliable

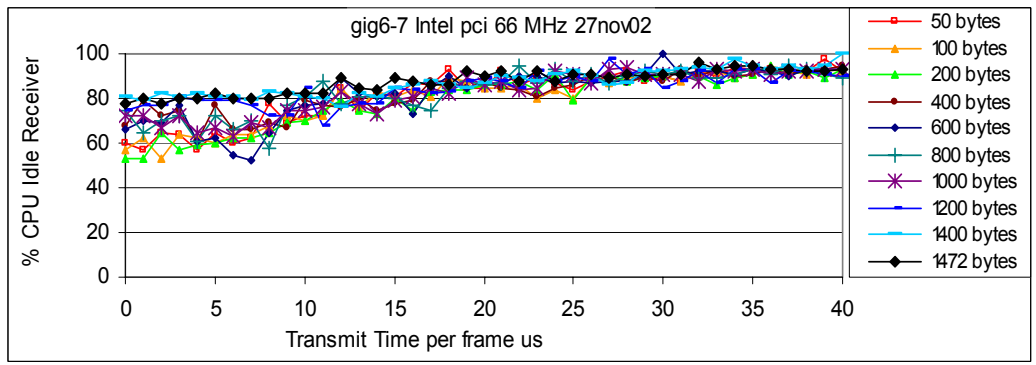
SuperMicro P4DP6: Throughput Intel Pro/1000

- Motherboard: SuperMicro P4DP6 Chipset: Intel E7500 (Plumas)
- CPU: Dual Xeon Prestonia 2.2 GHz **PCI, 64 bit, 66 MHz**
- RedHat 7.2 Kernel 2.4.19

- Max throughput **950Mbit/s**
- No packet loss

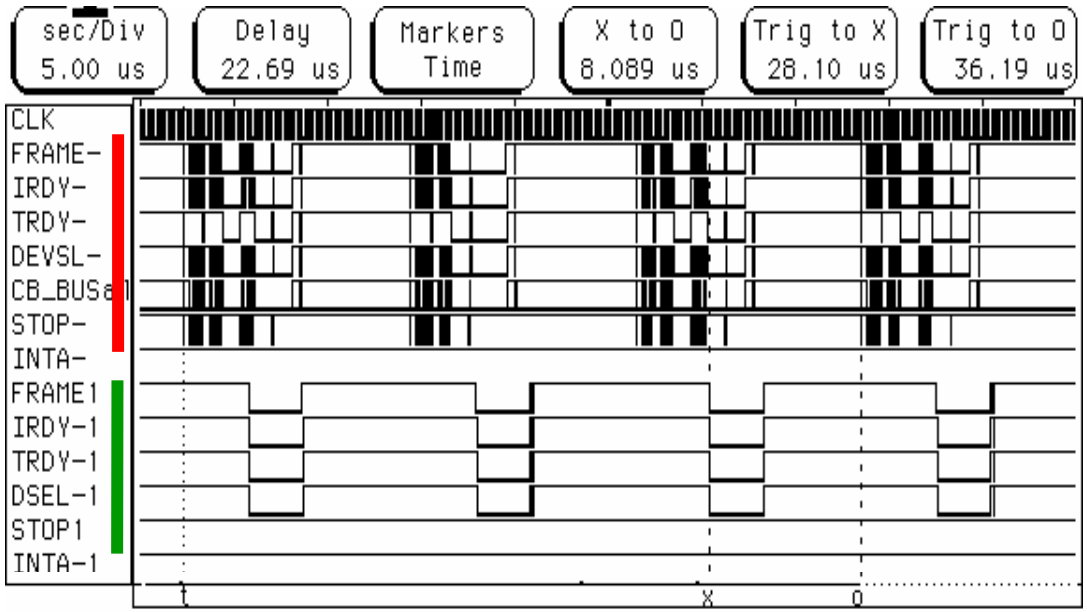


- Averages are misleading
- CPU utilisation on the receiving PC was ~ 25 % for packets > than 1000 bytes
- 30- 40 % for smaller packets

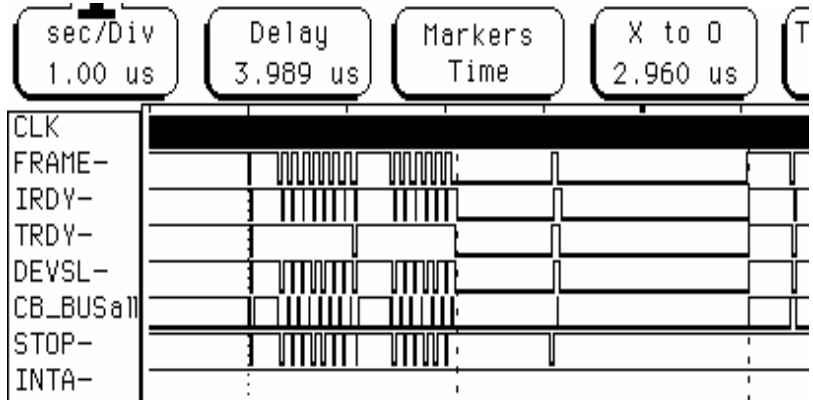


SuperMicro P4DP6: PCI Intel Pro/1000

- Motherboard: SuperMicro P4DP6 Chipset: Intel E7500 (Plumas)
- CPU: Dual Xeon Prestonia 2.2 GHz **PCI, 64 bit, 66 MHz**
- RedHat 7.2 Kernel 2.4.19



- 1400 bytes sent
- Wait 12 us
- ~5.14us on send PCI bus
- PCI bus ~68% occupancy
- ~ 3 us on PCI for data recv

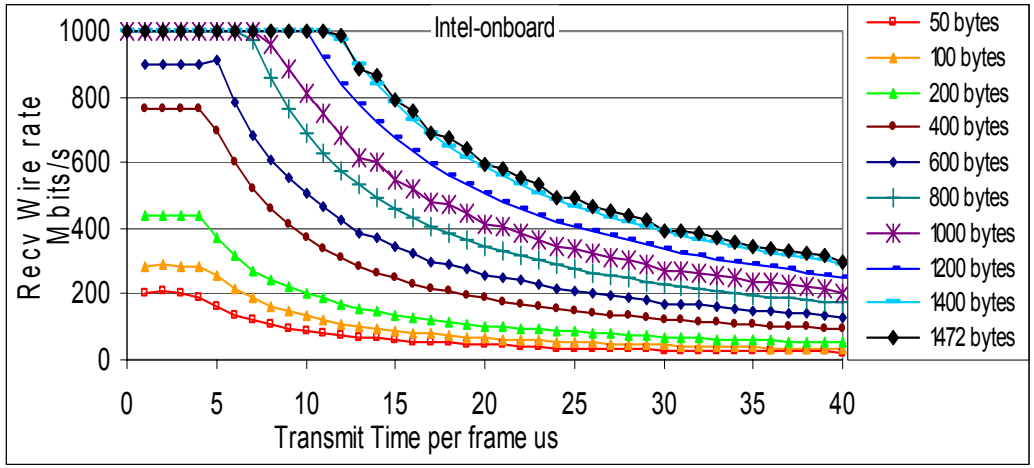


- CSR access inserts PCI STOPs
- NIC takes ~ 1 us/CSR
- CPU faster than the NIC !

- Similar effect with the SysKonnect NIC

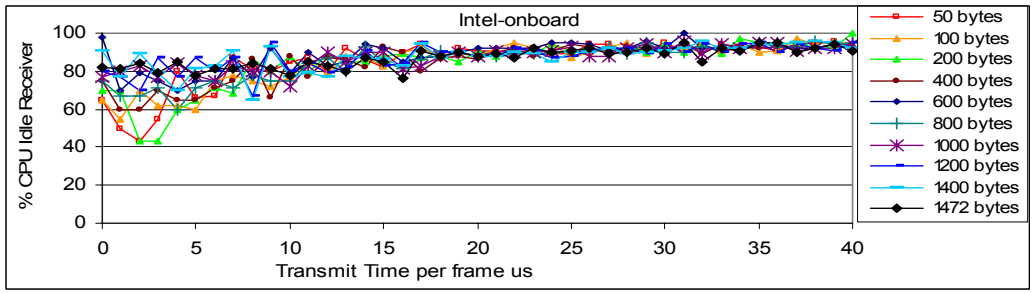
SuperMicro P4DP8-G2: Throughput Intel onboard

- Motherboard: SuperMicro P4DP8-G2 Chipset: Intel E7500 (Plumas)
- CPU: Dual Xeon Prestonia 2.4 GHz **PCI-X:64 bit**
- RedHat 7.3 Kernel 2.4.19



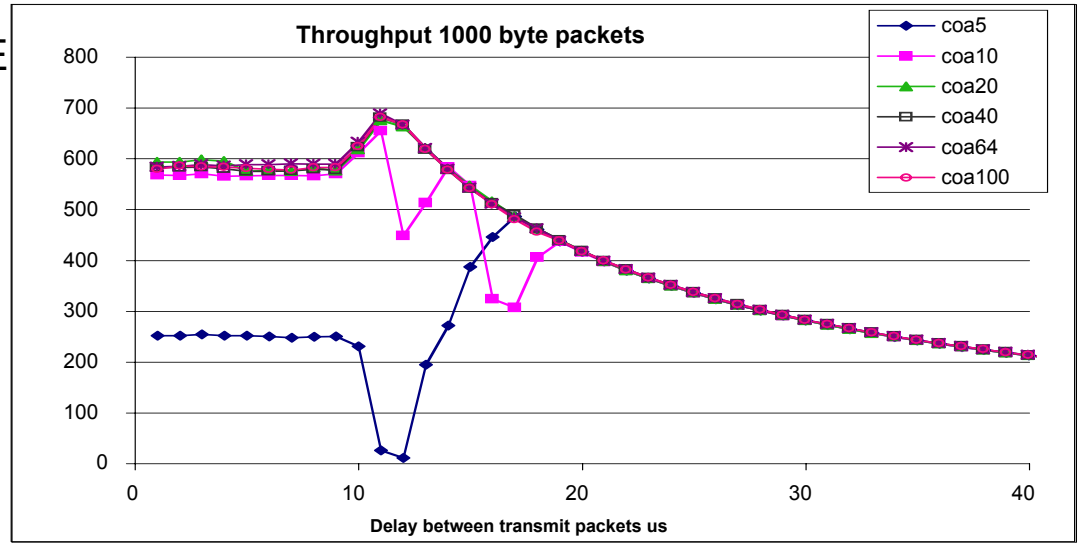
- Max throughput 995Mbit/s
- No packet loss

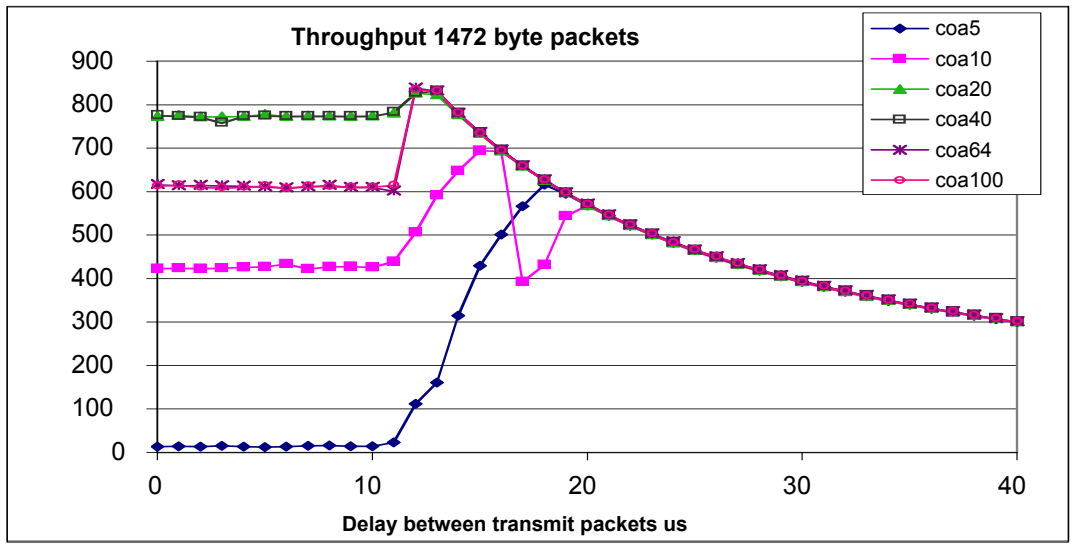
- Averages are misleading
- 20% CPU utilisation receiver packets > 1000 bytes
- 30% CPU utilisation smaller packets



Interrupt Coalescence: Throughput

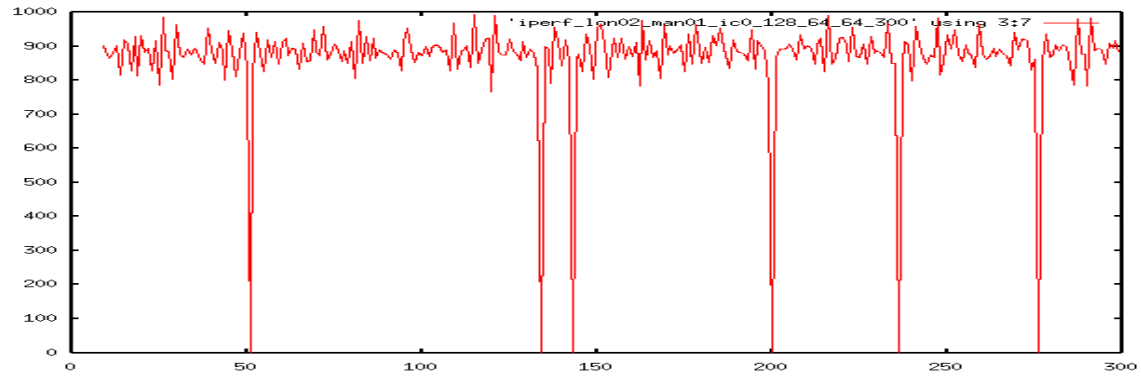
- Intel Pro 1000 on 370DLE



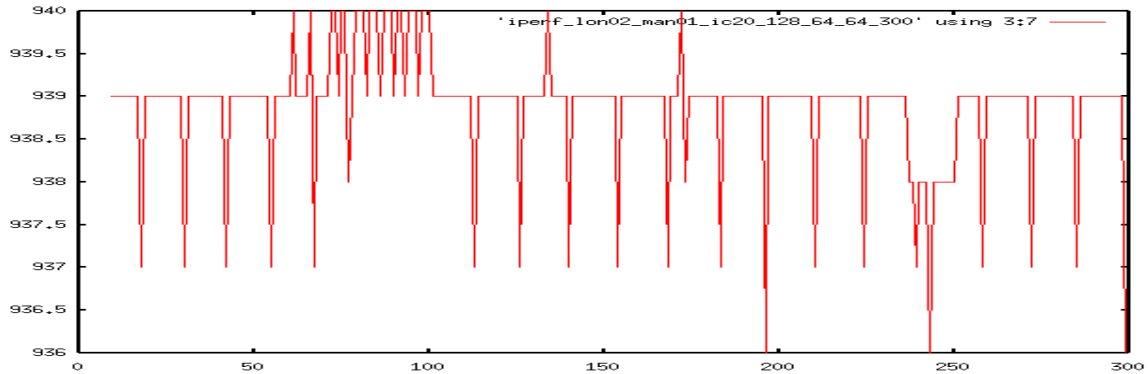


Interrupt Coalescence Investigations

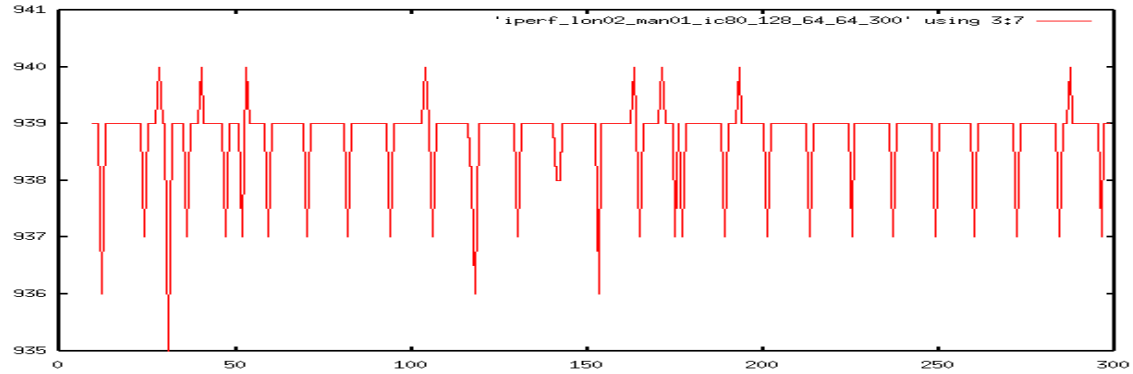
- **Set Kernel parameters for Socket Buffer size = rtt*BW**
- TCP mem-mem lon2-man1
- Tx 64 Tx-abs 64
- **Rx 0 Rx-abs 128**
- **820-980 Mbit/s +- 50 Mbit/s**



- Tx 64 Tx-abs 64
- **Rx 20 Rx-abs 128**
- **937-940 Mbit/s +- 1.5 Mbit/s**



- Tx 64 Tx-abs 64
- **Rx 80 Rx-abs 128**
- **937-939 Mbit/s +- 1 Mbit/s**



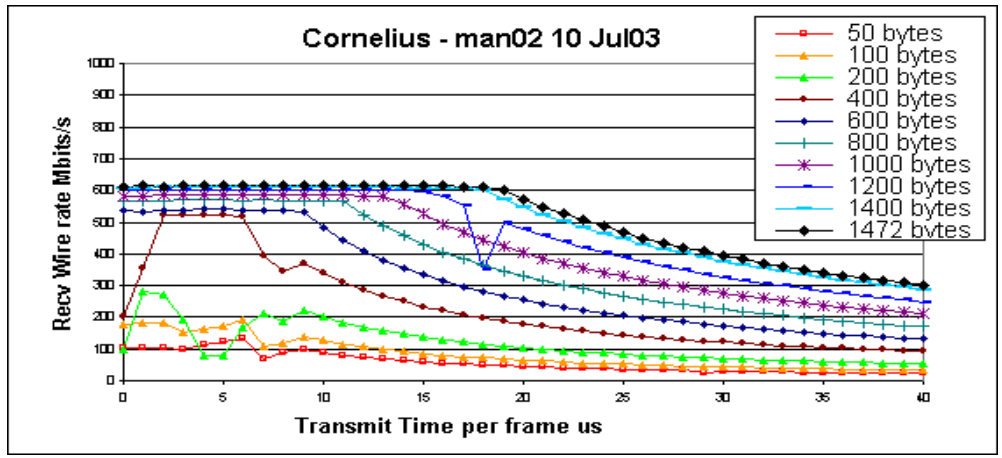
Supermarket Motherboards and other "Challenges"

Tyan Tiger S2466N

- **Motherboard: Tyan Tiger S2466N**
- **PCI: 1 64bit 66 MHz**
- **CPU: Athlon MP2000+**
- **Chipset: AMD-760 MPX**

- **3Ware forces PCI bus to 33 MHz**

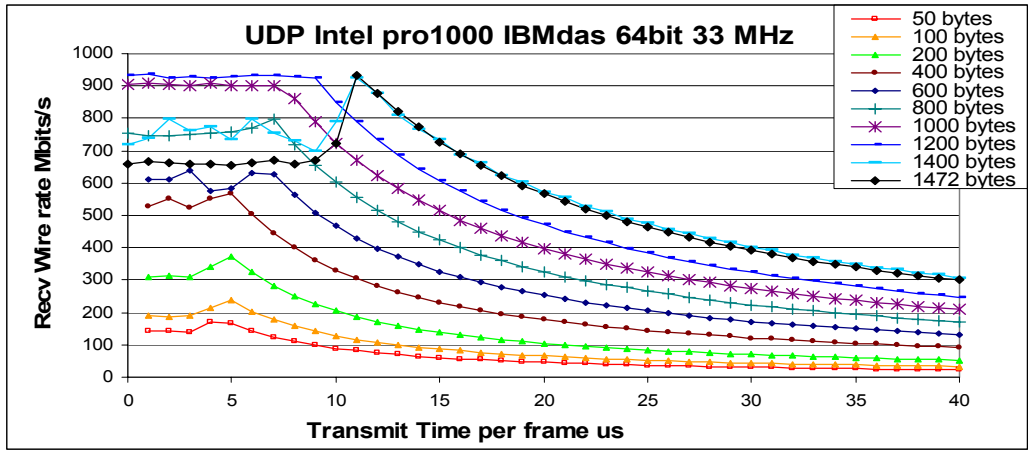
- Tyan to MB-NG SuperMicro
  Network mem-mem 619 Mbit/s



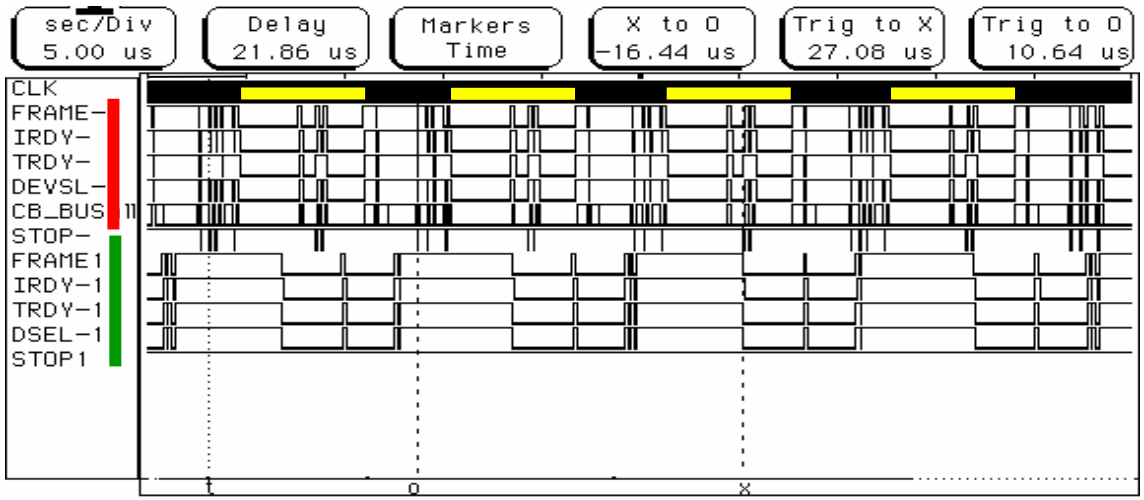
IBM das: Throughput: Intel Pro/1000

- Motherboard: IBM das Chipset:: ServerWorks CNB20LE
- CPU: Dual PIII 1GHz **PCI:64 bit 33 MHz**
- RedHat 7.1 Kernel 2.4.14

- Max throughput 930Mbit/s
- No packet loss > 12 us
- Clean behaviour
- Packet loss during drop

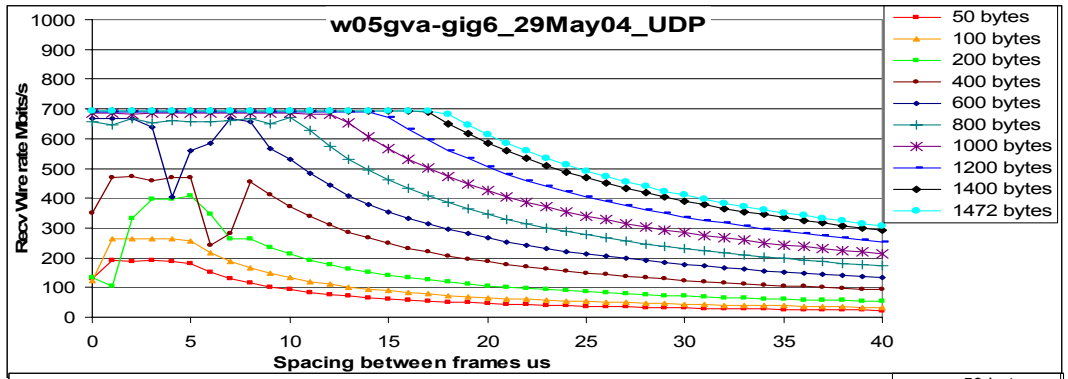


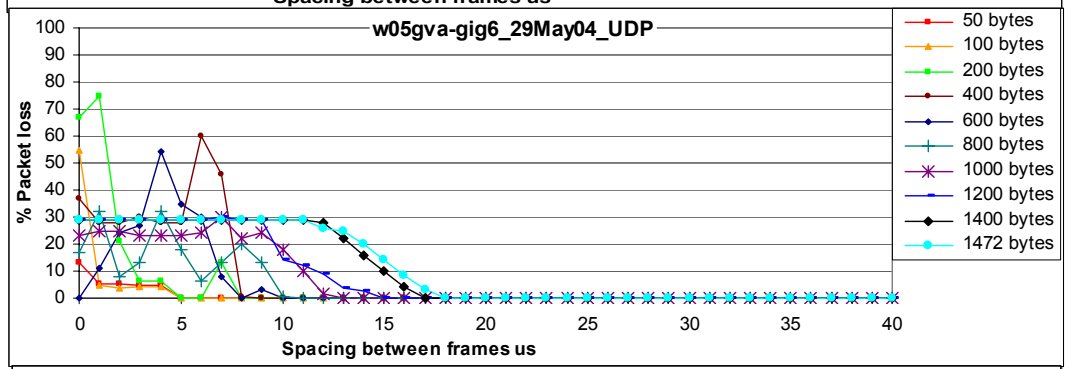
- 1400 bytes sent
- 11 us spacing
- Signals clean
- ~9.3us on send PCI bus
- **PCI bus ~82% occupancy**
- ~ 5.9 us on PCI for data recv.

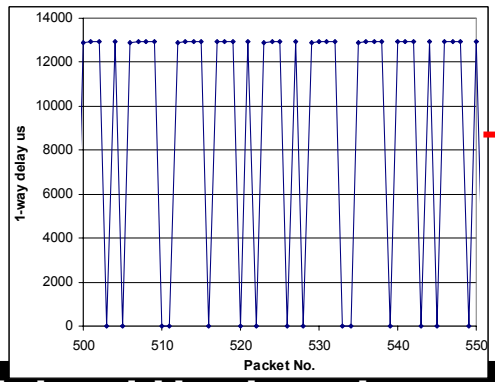


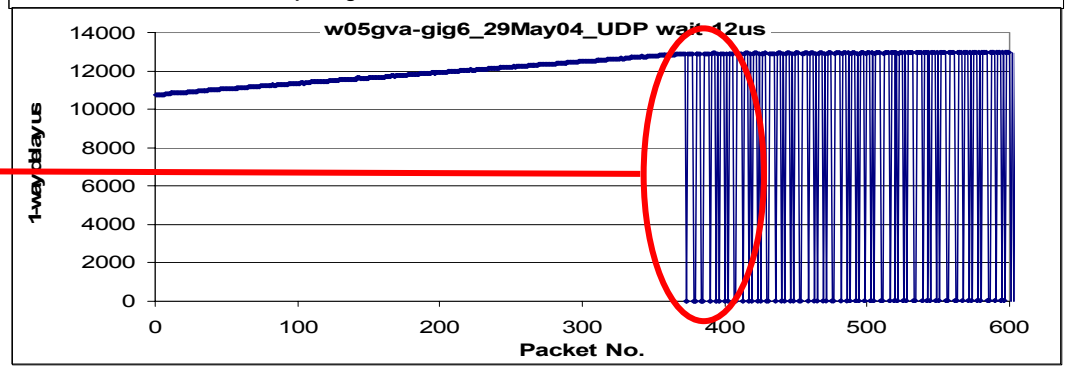
Network switch limits behaviour

- End2end UDP packets from udpmon
  - Only 700 Mbit/s throughput
  - Lots of packet loss
  - Packet loss distribution shows throughput limited





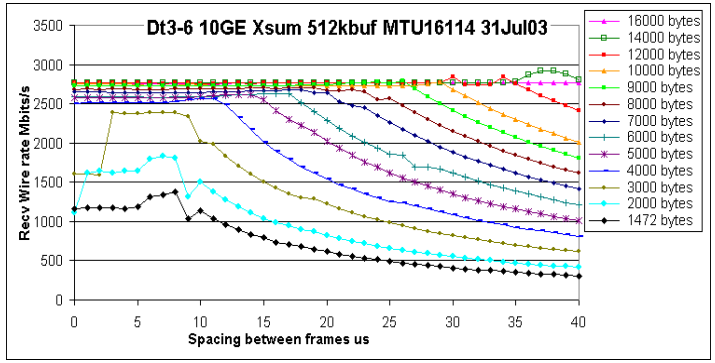




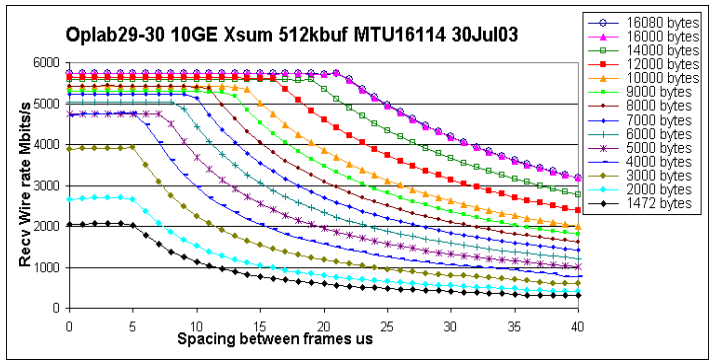
10 Gigabit Ethernet

10 Gigabit Ethernet: UDP Throughput

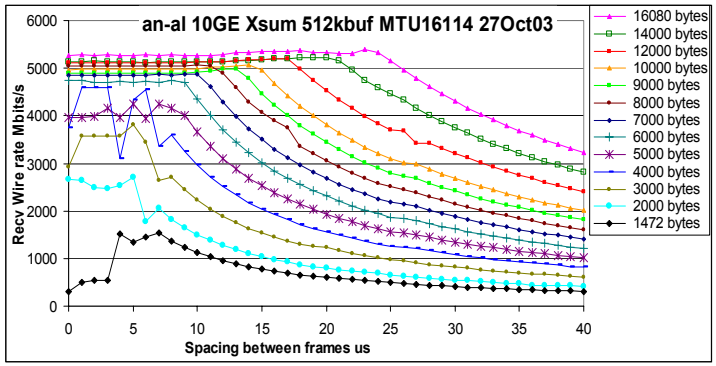
- 1500 byte MTU gives ~ 2 Gbit/s
- Used 16144 byte MTU max user length 16080
- DataTAG Supermicro PCs
- Dual 2.2 GHz Xenon CPU FSB 400 MHz
- PCI-X mmrbc 512 bytes
- wire rate throughput of 2.9 Gbit/s



- CERN OpenLab HP Itanium PCs
- Dual 1.0 GHz 64 bit Itanium CPU FSB 400 MHz
- PCI-X mmrbc 4096 bytes
- wire rate of 5.7 Gbit/s

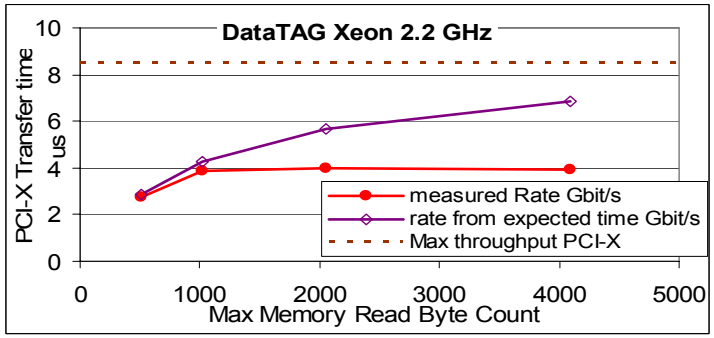


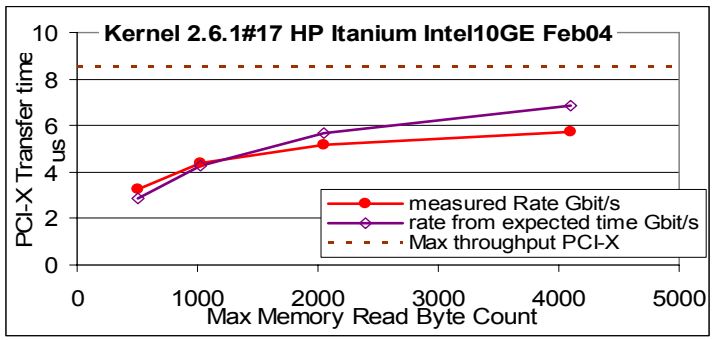
- SLAC Dell PCs giving a
- Dual 3.0 GHz Xenon CPU FSB 533 MHz
- PCI-X mmrbc 4096 bytes
- wire rate of 5.4 Gbit/s

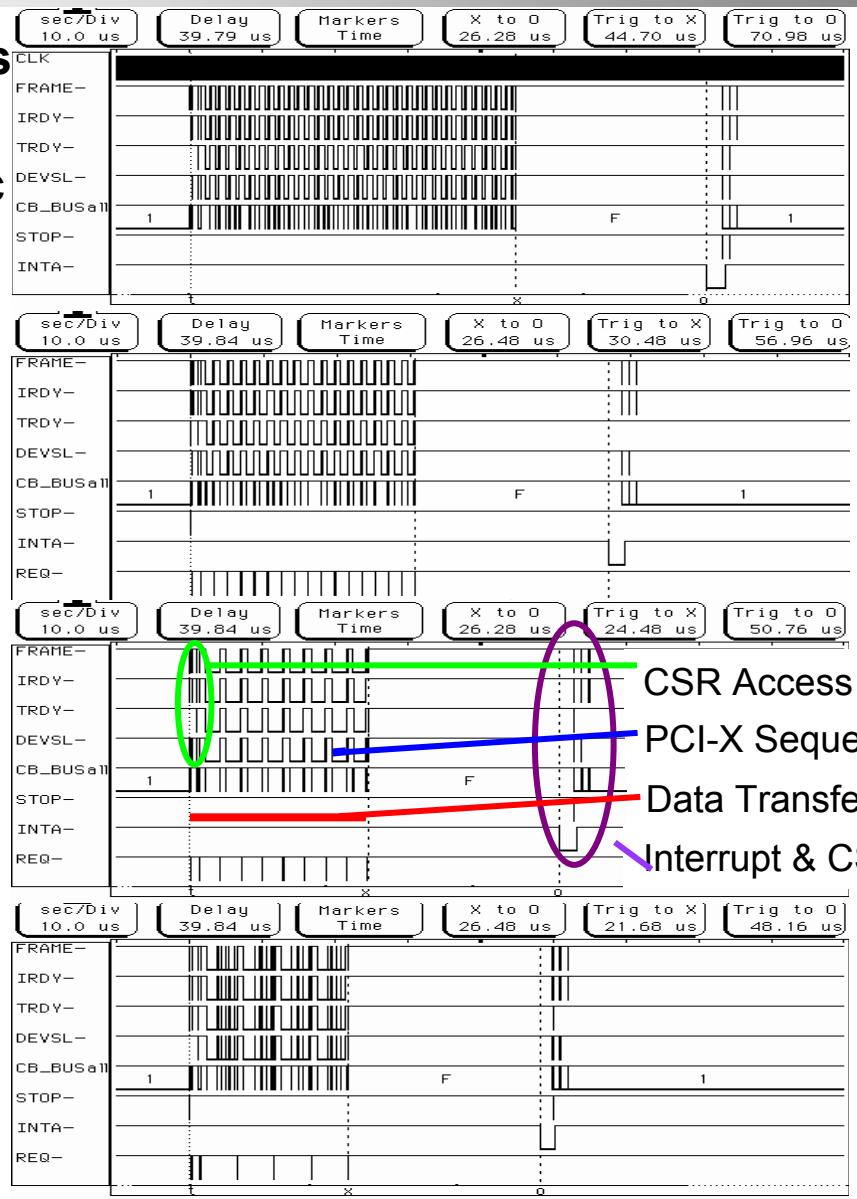


10 Gigabit Ethernet: Tuning PCI-X

- 16080 byte packets every 200 µs
- Intel PRO/10GbE LR Adapter
- PCI-X bus occupancy vs mmrbc
  - Measured times
  - Times based on PCI-X times from the logic analyser
  - Expected throughput ~7 Gbit/s
  - Measured 5.7 Gbit/s







Different TCP Stacks

Investigation of new TCP Stacks

- **The AIMD Algorithm – Standard TCP (Reno)**
  - For each ack in a RTT **without** loss:

  cwnd -> cwnd + a / cwnd ***- Additive Increase, a=1***
  - For each window **experiencing loss**:

  cwnd -> cwnd – b (cwnd) ***- Multiplicative Decrease, b*** = ½
- **High Speed TCP**

a and **b** vary depending on **current cwnd** using a table
  - **a** increases more rapidly with larger cwnd – returns to the 'optimal' cwnd size sooner for the network path
  - **b** decreases less aggressively and, as a consequence, so does the cwnd. The effect is that there is not such a decrease in throughput.
- **Scalable TCP**

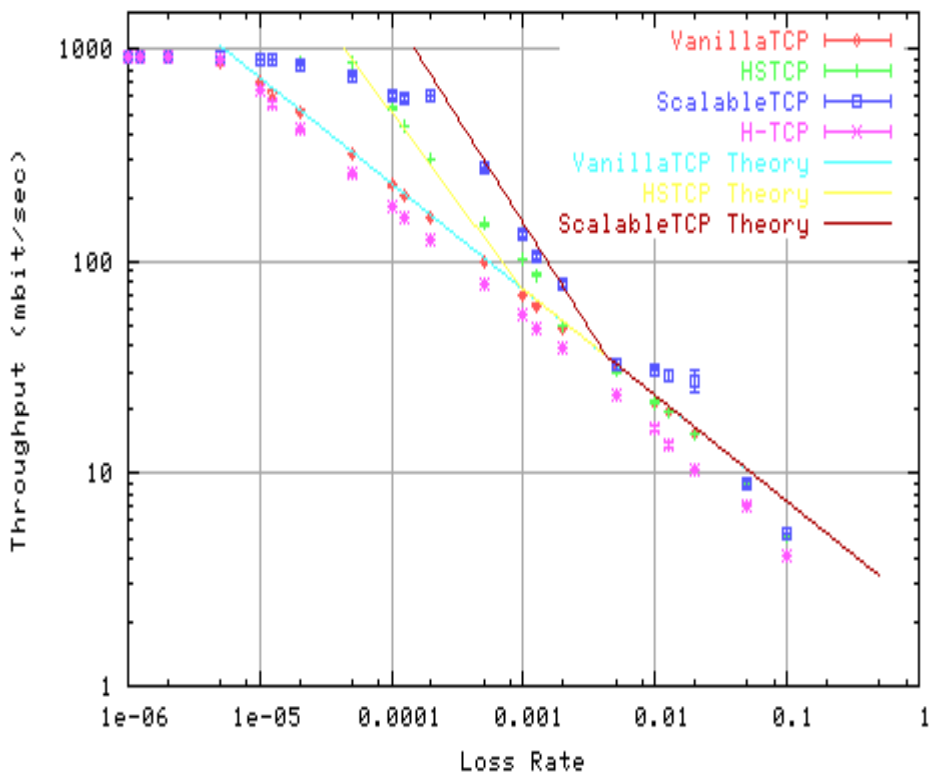
a and **b** are **fixed** adjustments for the increase and decrease of cwnd
  - **a** = 1/100 – the increase is greater than TCP Reno
  - **b** = 1/8 – the decrease on loss is less than TCP Reno
  - Scalable over any link speed.
- **Fast TCP**

Uses round trip time as well as packet loss to indicate congestion with rapid convergence to fair equilibrium for throughput.

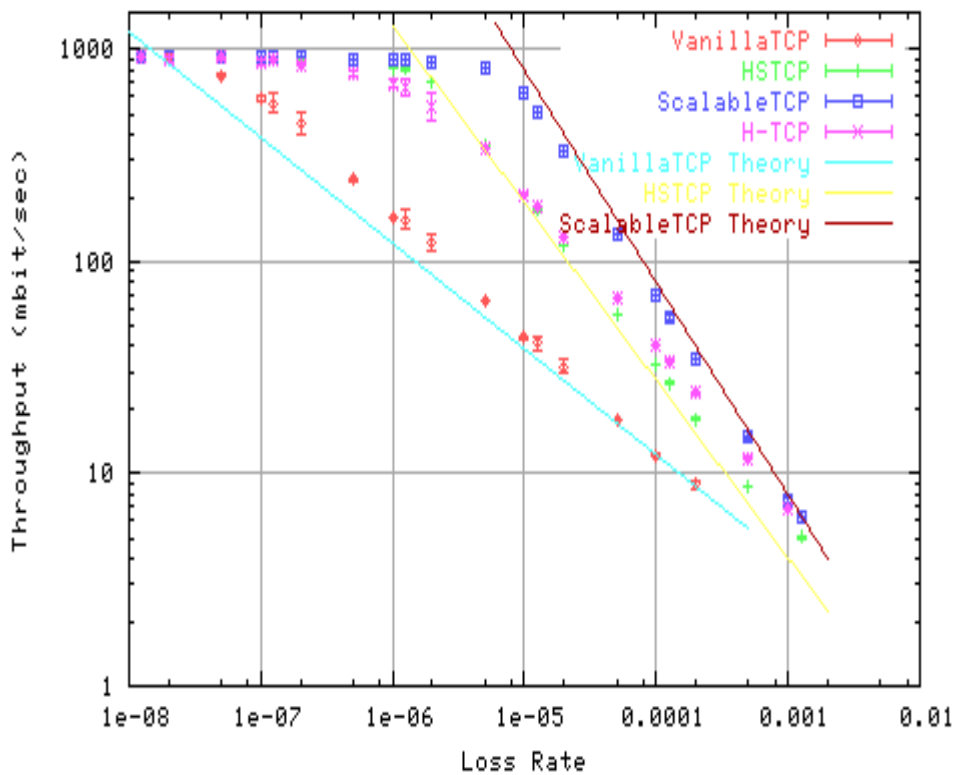
Comparison of TCP Stacks

- TCP Response Function
  - Throughput vs Loss Rate – further to right: faster recovery
  - Drop packets in kernel

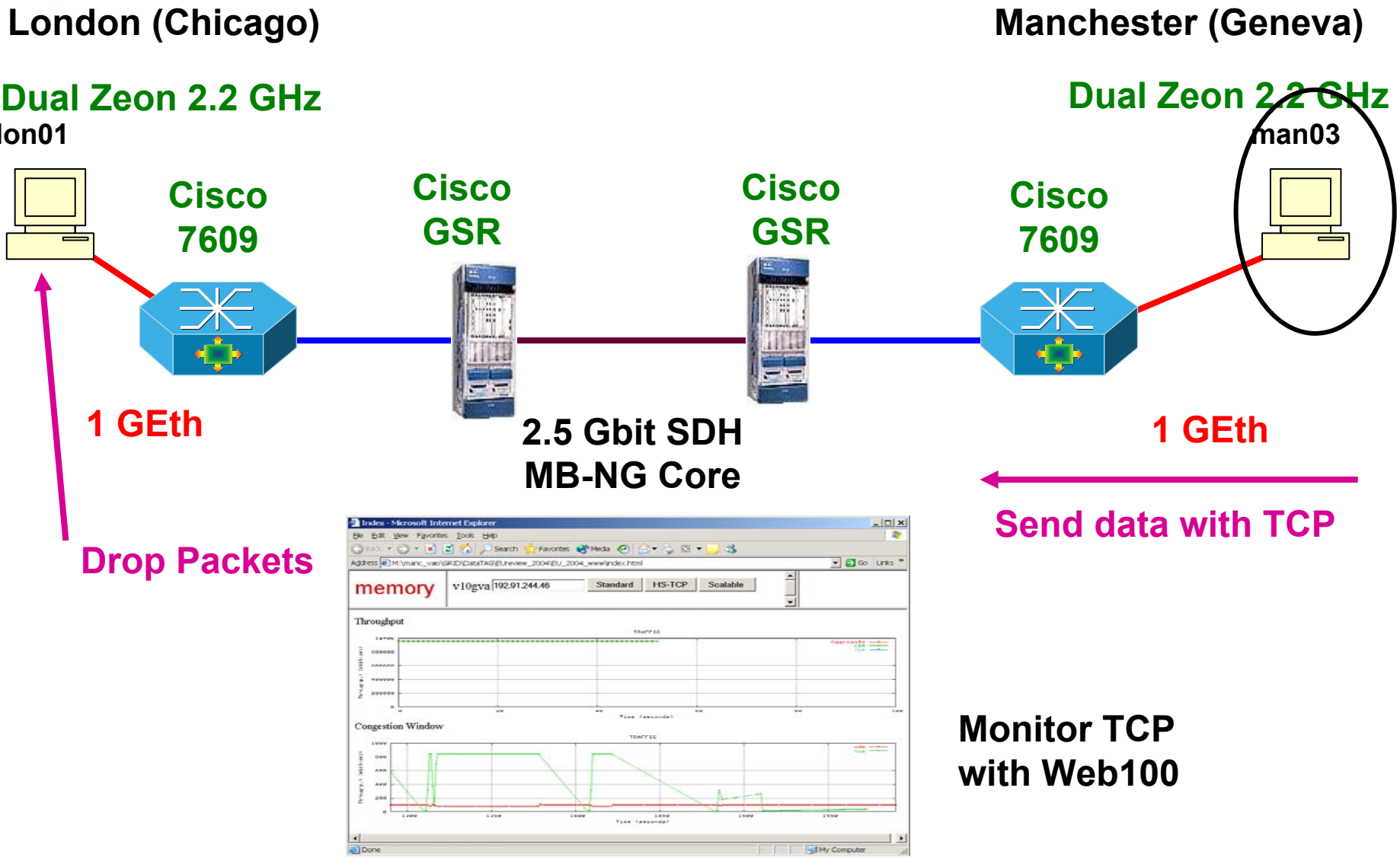
MB-NG rtt 6ms



DataTAG rtt 120 ms

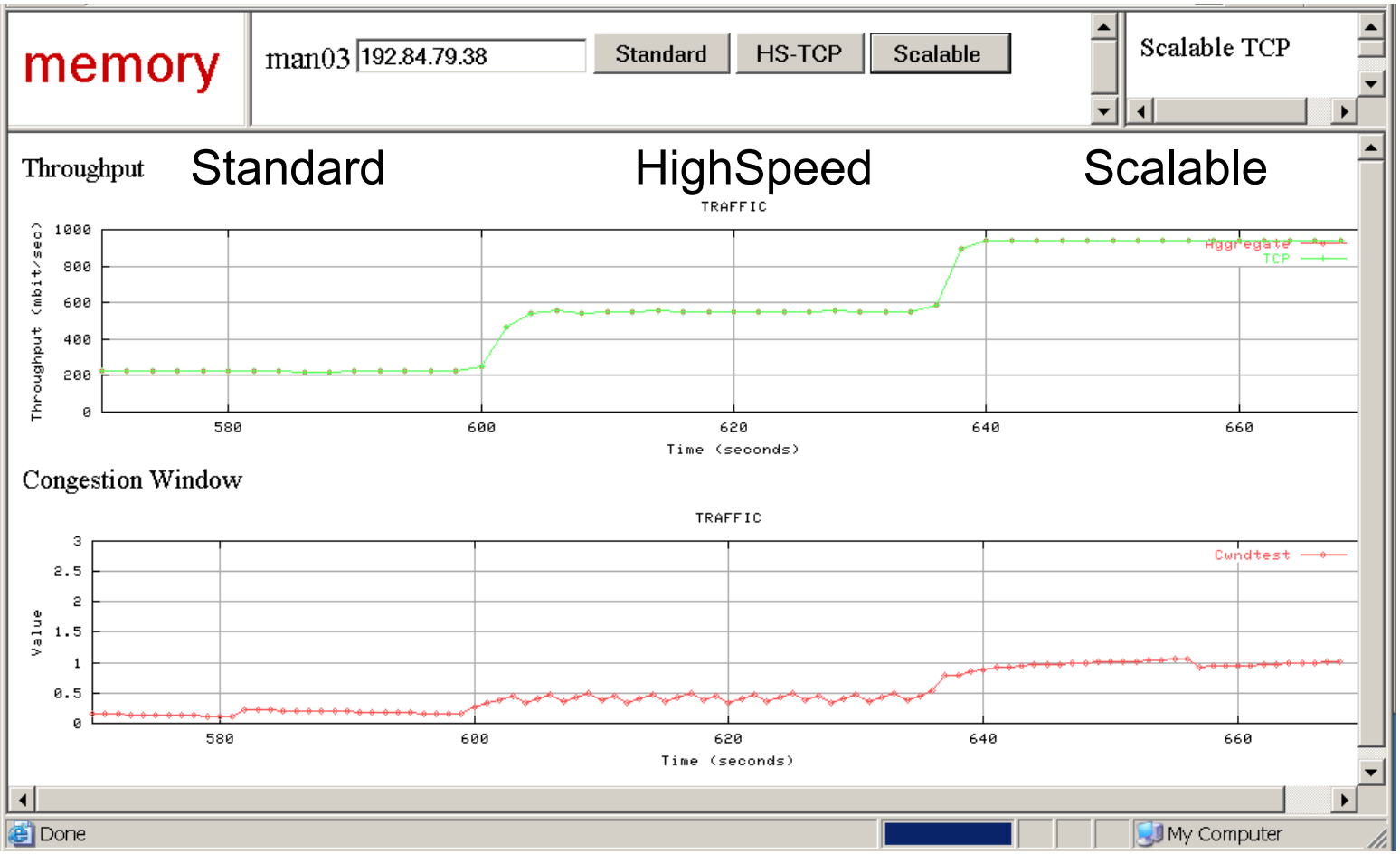


High Throughput Demonstrations



High Performance TCP – MB-NG

- Drop 1 in 25,000
- rtt 6.2 ms
- Recover in 1.6 s



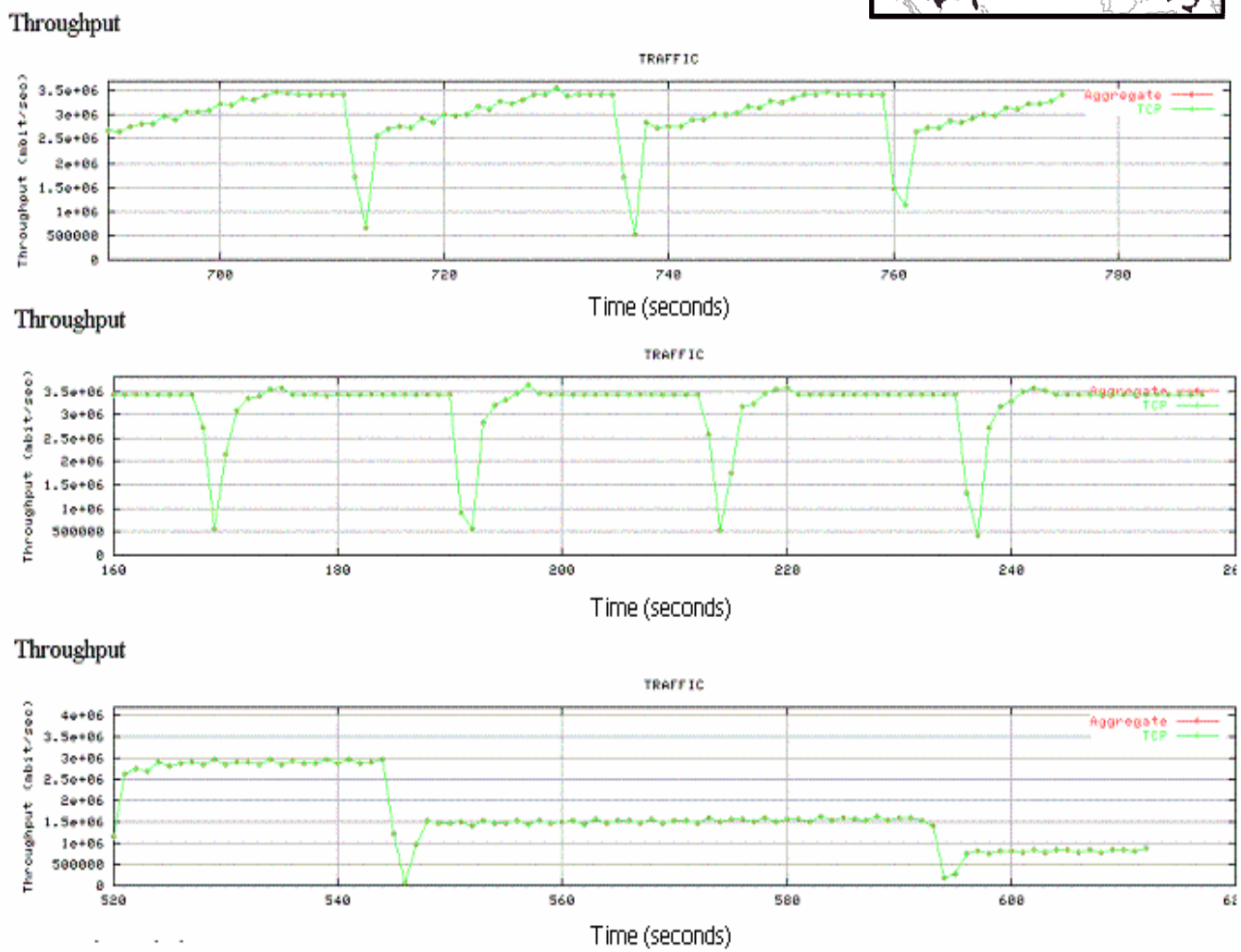
High Performance TCP – DataTAG

- Different TCP stacks tested on the DataTAG Network
- rtt 128 ms
- Drop 1 in 10^6

- High-Speed
  - Rapid recovery

- Scalable
  - Very fast recovery

- Standard
  - Recovery would take ~ 20 mins



Disk and RAID Sub-Systems

Based on talk at NFNN 2004

by

Andrew Sansum

Tier 1 Manager at RAL

Reliability and Operation

- The performance of a system that is broken is 0.0 MB/S!
- When staff are fixing broken servers they cannot spend their time optimising performance.
- With a lot of systems, anything that can break – will break
  - Typical ATA failure rate 2-3% per annum at RAL, CERN and CASPUR. That's 30-45 drives per annum on the RAL Tier1 **i.e. One/week**. One failure for every 10**15 bits read. MTBF 400K hours.
  - RAID arrays may not handle block re-maps satisfactorily or take so long that the system has given up.
    RAID5 not perfect protection – finite chance of 2 disks failing!
  - SCSI interconnects corrupts data or give CRC errors
  - Filesystems (ext2) corrupt for no apparent cause (EXT2 errors)
  - Silent data corruption happens (checksums are vital).
- Fixing data corruptions can take a huge amount of time (> 1 week). Data loss possible.

You need to Benchmark – but how?

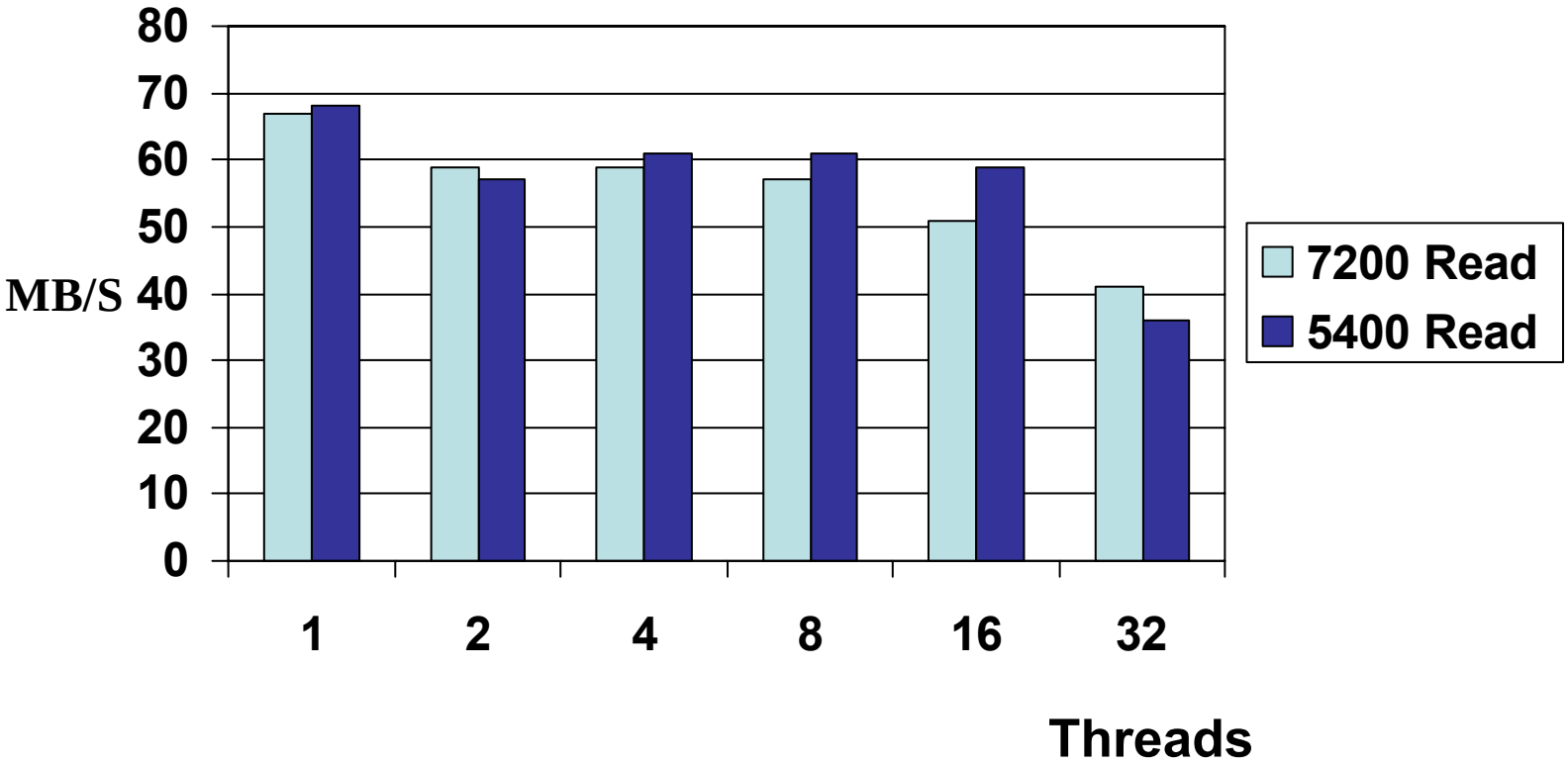
- Andrew's golden rule: Benchmark Benchmark Benchmark
- Choose a benchmark that matches your application (we use Bonnie++ and IOZONE).
  - Single stream of sequential I/O.
  - Random disk accesses.
  - Multi-stream – 30 threads more typical.
- Watch out for caching effects (use large files and small memory).
- Use a standard protocol that gives reproducible results.
- Document what you did for future reference
- Stick with the same benchmark suite/protocol as long as possible to gain familiarity. Much easier to spot significant changes

Hard Disk Performance

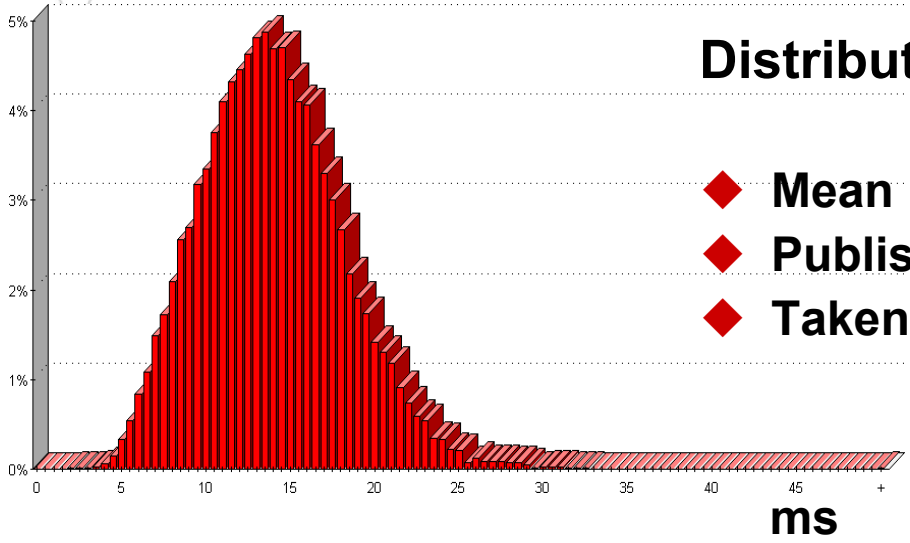
- Factors affecting performance:
  - Seek time (rotation speed is a big component)
  - Transfer Rate
  - Cache size
  - Retry count (vibration)
- Watch out for unusual disk behaviours:
  - write/verify (drive verifies writes for first <n> writes)
  - drive spin down etc etc.
- Storage review is often worth reading: http://www.storagereview.com/

Impact of Drive Transfer Rate

A slower drive may not cause significant sequential performance degradation in a RAID array. This 5400 drive was 25% slower than the 7200



Disk Parameters

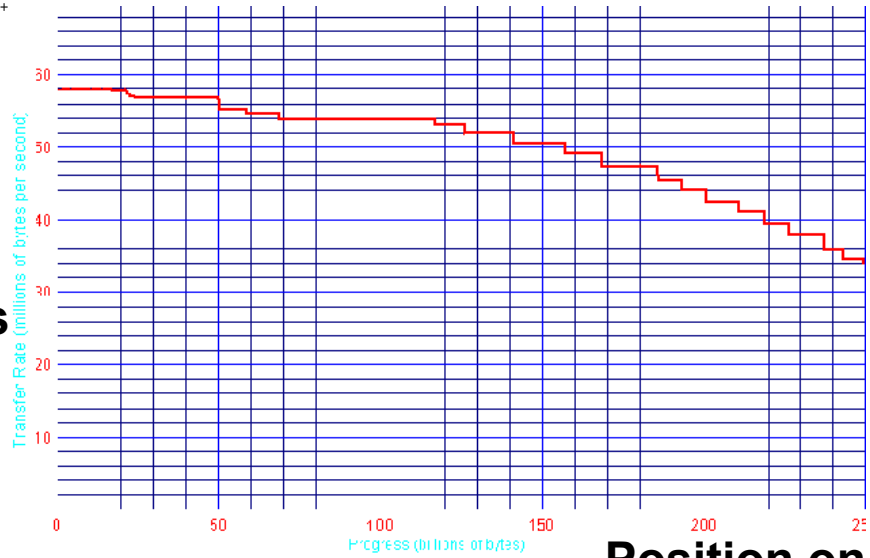


Distribution of Seek Time

- Mean 14.1 ms
- Published tseek time 9 ms
- Taken off rotational latency

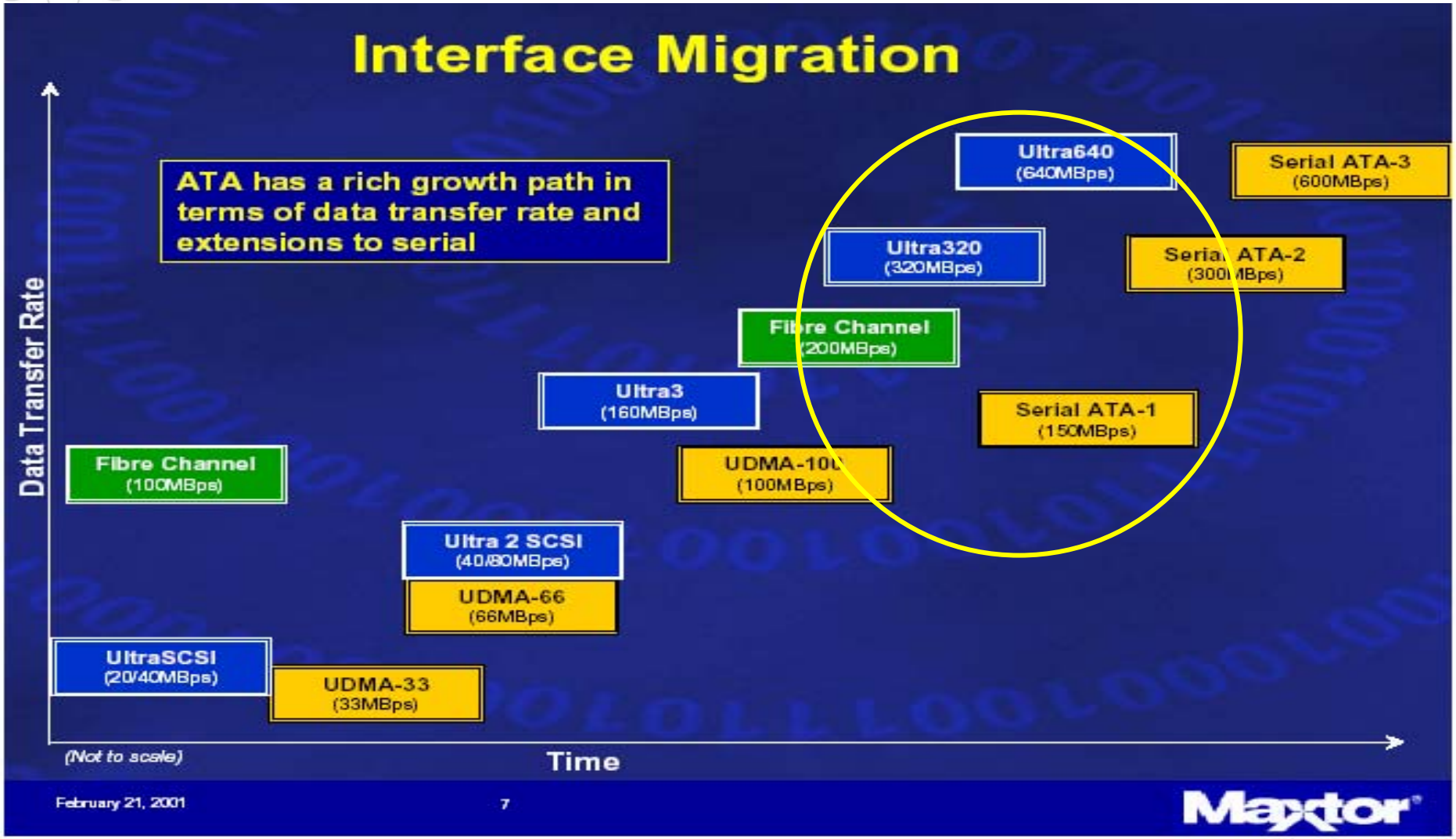
Head Position & Transfer Rate

- Outside of disk has more blocks so faster
- 58 MB/s to 35 MB/s
- 40% effect



From: www.storagereview.com

Drive Interfaces



http://www.maxtor.com/

RAID levels

- Choose Appropriate RAID level
  - RAID 0 (stripe) is fastest but no redundancy
  - RAID 1 (mirror) single disk performance
  - RAID 2, 3 and 4 not very common/supported sometimes worth testing with your application
  - RAID 5 – Read is almost as fast as RAID-0, write substantially slower.
  - RAID 6 – extra parity info – good for unreliable drives but could have considerable impact on performance. Rare but potentially very useful for large IDE disk arrays.
  - RAID 50 & RAID 10 - RAID0 2 (or more) controllers

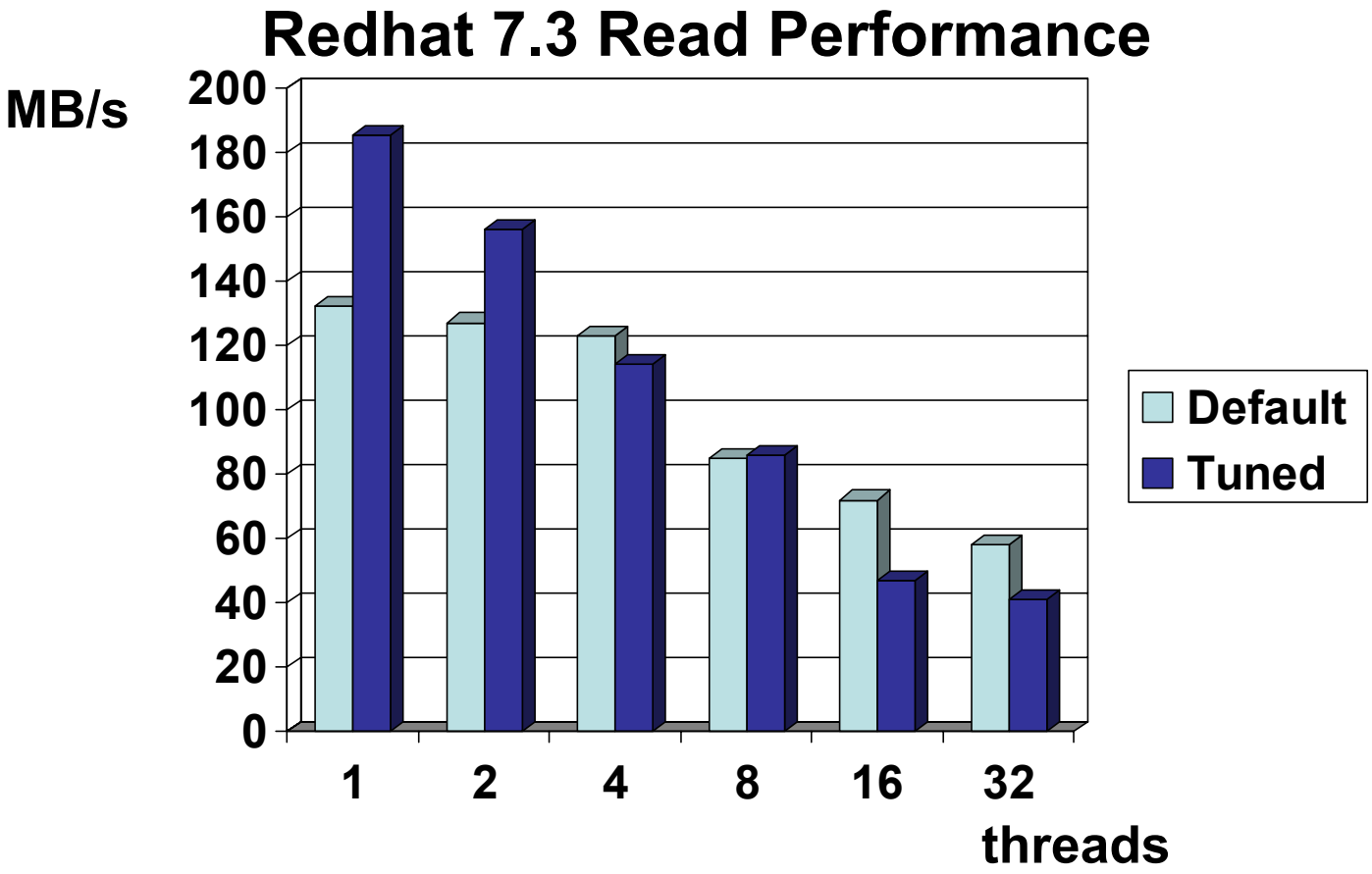
Kernel Versions

- Kernel versions can make an enormous difference. It is not unusual for I/O to be badly broken in Linux.
  - In 2.4 series – Virtual Memory Subsystem problems
    - only 2.4.5 some 2.4.9 >2.4.17 OK
  - Redhat Enteprise 3 seems badly broken (although most recent RHE Kernel may be better)
  - 2.6 kernel contains new functionality and may be better (although first tests show little difference)
- Always check after an upgrade that performance remains good.

Kernel Tuning

- Tunable kernel parameters can improve I/O performance, but depends what you are trying to achieve.
  - IRQ balancing. Issues with IRQ handling on some Xeon chipsets/kernel combinations (all IRQs handled on CPU zero).
  - Hyperthreading
  - I/O schedulers – can tune for latency by minimising the seeks
    - /sbin/elvtune (number sectors of writes before reads allowed)
    - Choice of scheduler: elevator=xx orders I/O to minimise seeks, merges adjacent requests.
  - VM on 2.4 tuningBdflush and readahead (single thread helped)
    - Sysctl –w vm.max-readahead=512
    - Sysctl –w vm.min-readahead=512
    - Sysctl –w vm.bdflush=10 500 0 0 3000 10 20 0
  - On 2.6 readahead command is [default 256]
    - blockdev --setra 16384 /dev/sda

Example Effect of VM Tuneup

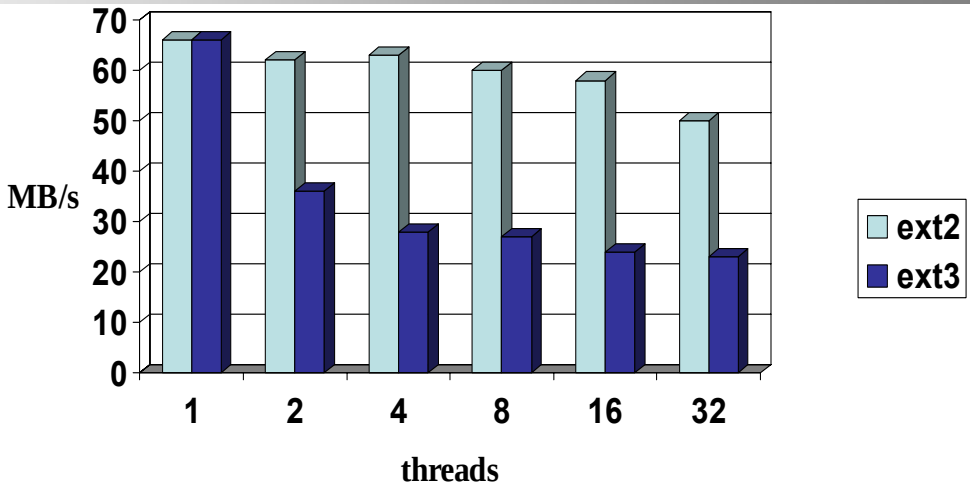


g.prassas@rl.ac.uk

IOZONE Tests

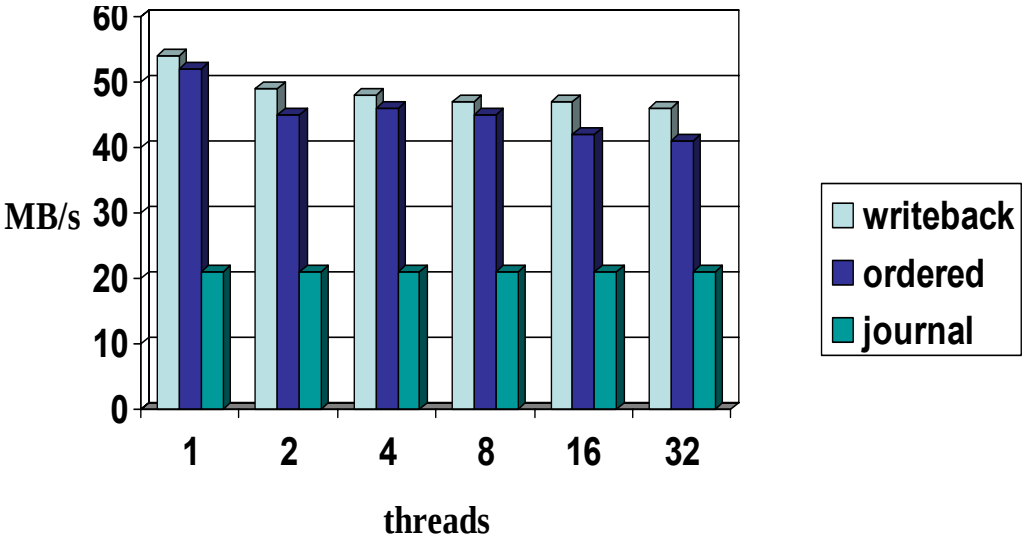
Read Performance

- File System ext2 ext3
- Due to journal behaviour
- 40% effect

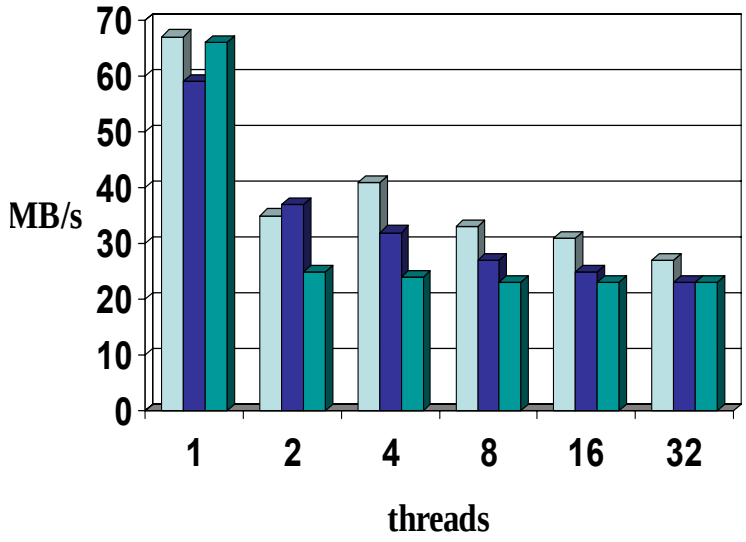


ext3 Write

- Journal write data + write journal



ext3 Read

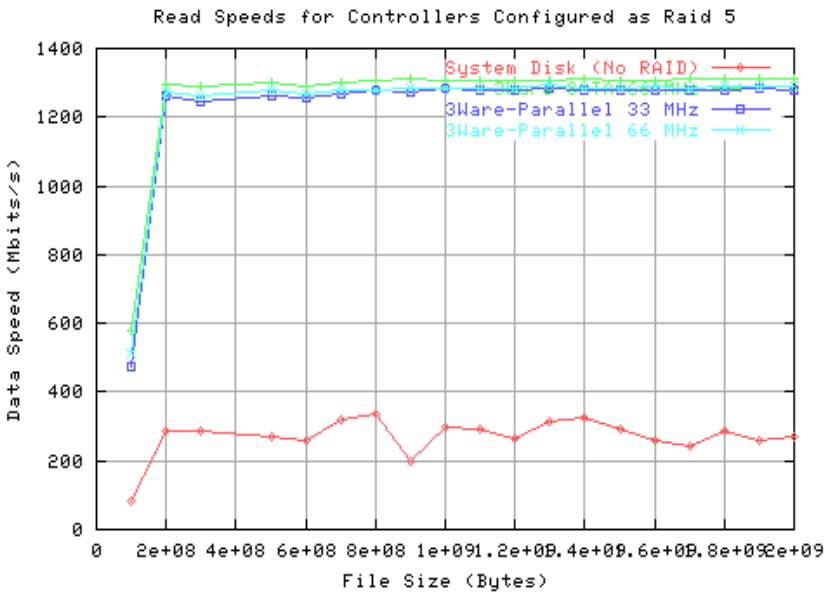


RAID Controller Performance

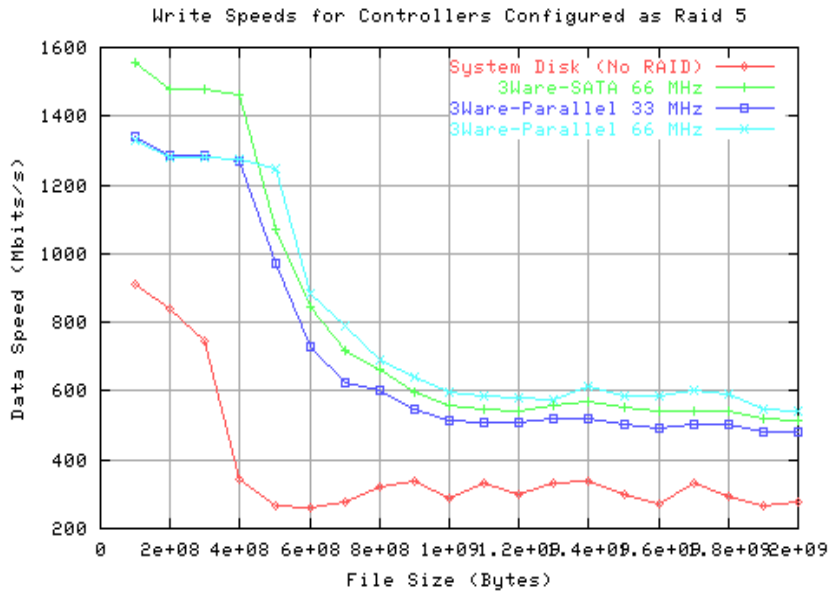
Stephen Dallison

- **RAID5** (stripped with redundancy)
- 3Ware 7506 Parallel 66 MHz
- 3Ware 7505 Parallel 33 MHz
- 3Ware 8506 Serial ATA 66 MHz
- ICP Serial ATA 33/66 MHz
- Tested on Dual 2.2 GHz Xeon Supermicro P4DP8-G2 motherboard
- Disk: Maxtor 160GB 7200rpm 8MB Cache
- Read ahead kernel tuning: /proc/sys/vm/max-readahead = 512

Disk – Memory Read Speeds



Memory - Disk Write Speeds

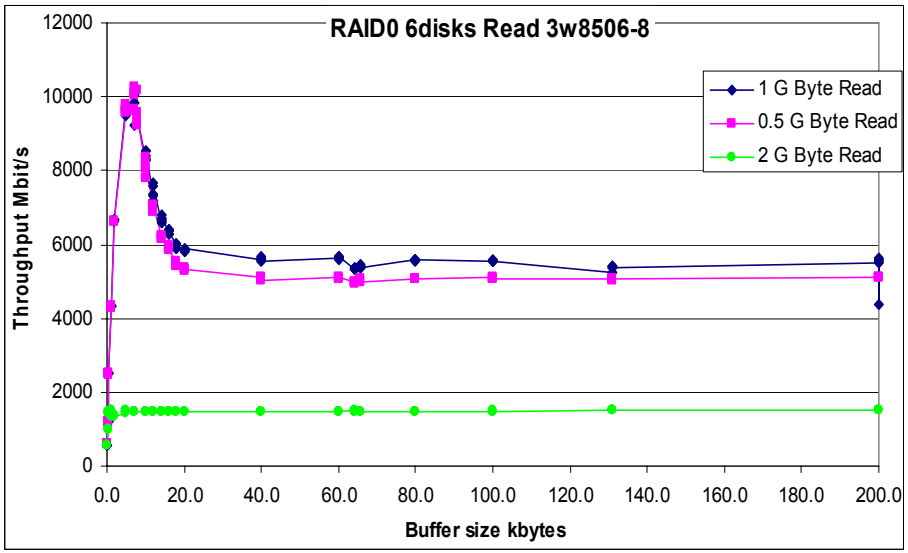


- Rates for the same PC RAID0 (stripped) Read 1040 Mbit/s, Write 800 Mbit/s

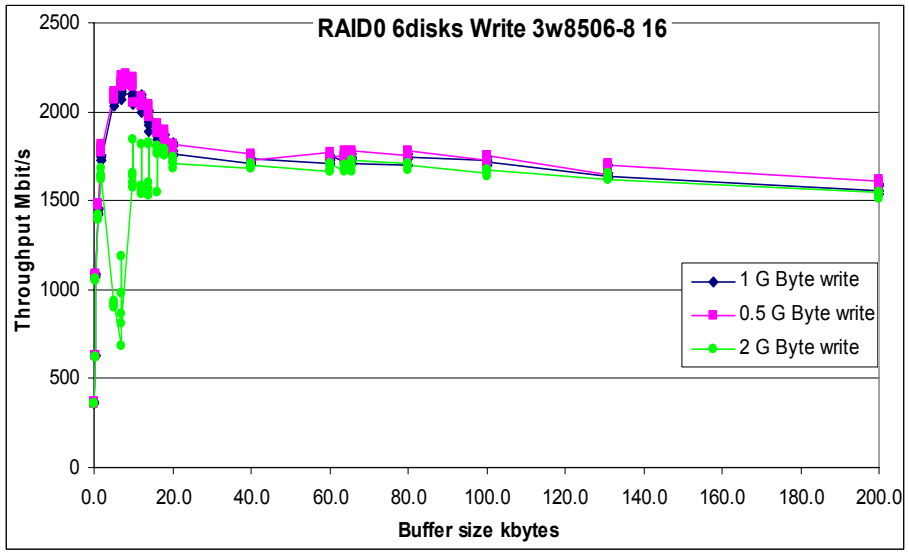
SC2004 RAID Controller Performance

- Supermicro X5DPE-G2 motherboards loaned from Boston Ltd.
- Dual 2.8 GHz Zeon CPUs with 512 k byte cache and 1 M byte memory
- 3Ware 8506-8 controller on 133 MHz PCI-X bus
- Configured as **RAID0** 64k byte stripe size
- Six 74.3 GByte Western Digital Raptor WD740 SATA disks
  - 75 Mbyte/s disk-buffer 150 Mbyte/s buffer-memory
- Scientific Linux with 2.6.6 Kernel + altAIMD patch (Yee) + packet loss patch
- Read ahead kernel tuning: /sbin/blockdev --setra 16384 /dev/sda

Disk – Memory Read Speeds



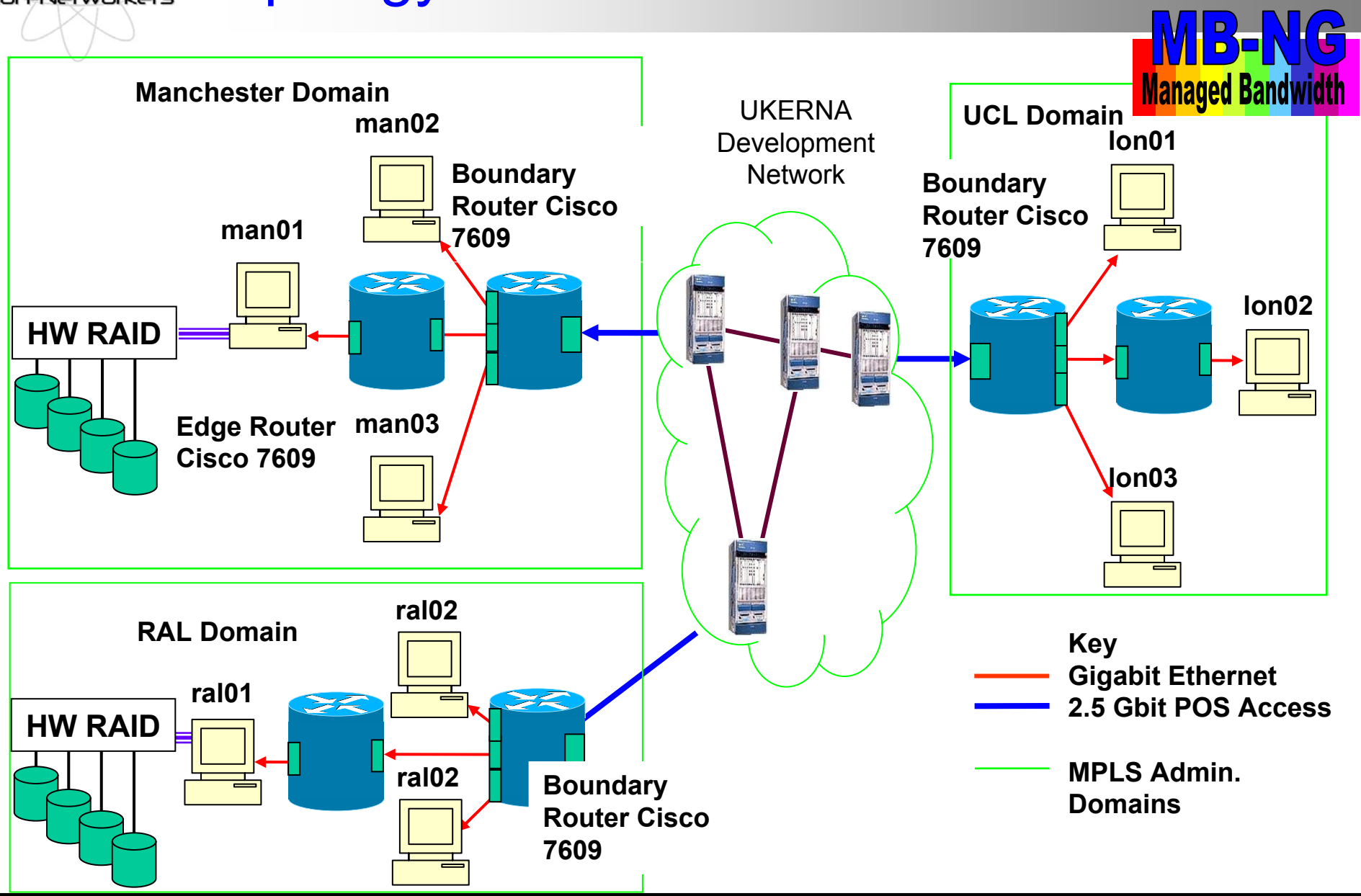
Memory - Disk Write Speeds



- RAID0 (stripped) 2 GByte file: Read 1500 Mbit/s, Write 1725 Mbit/s

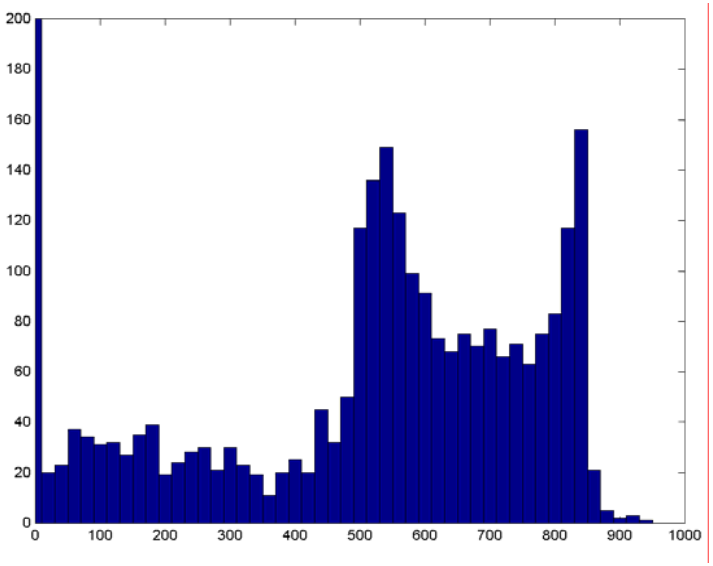
Applications
Throughput for Real Users

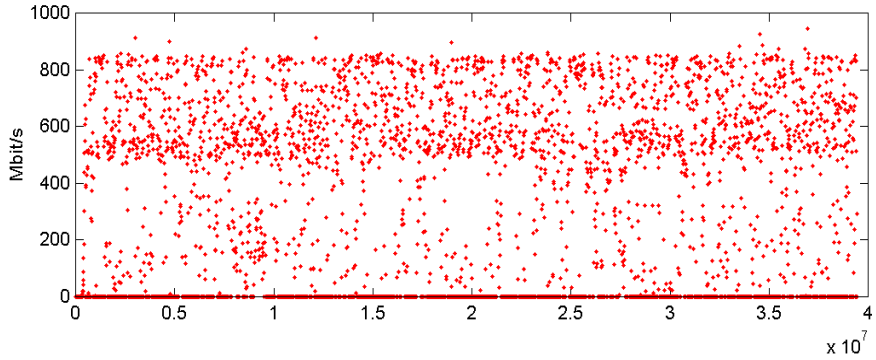
Topology of the MB – NG Network

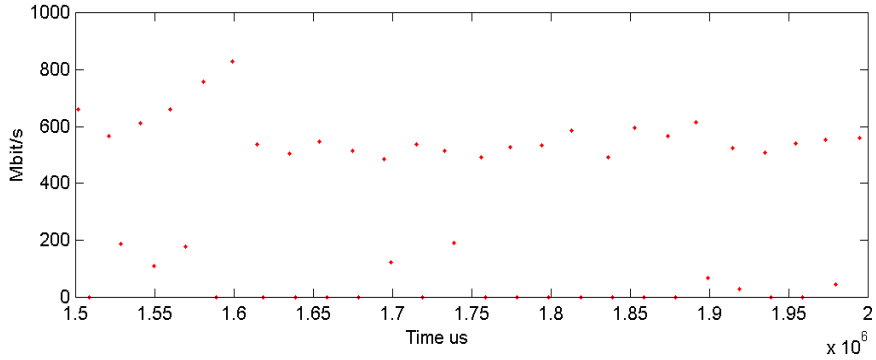


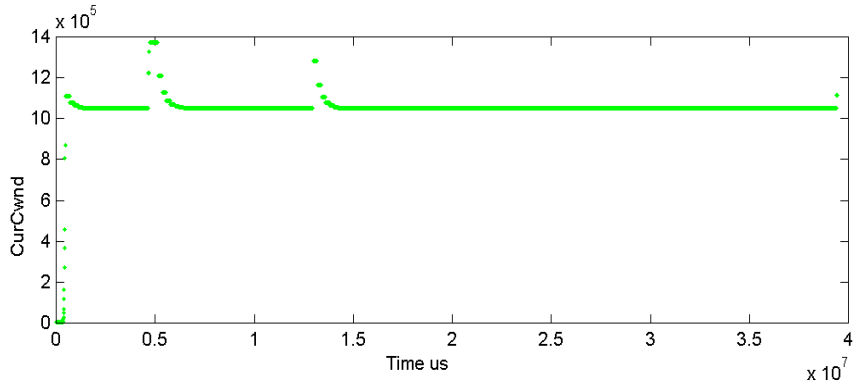
Gridftp Throughput + Web100

- **RAID0 Disks:**
  - **960 Mbit/s read**
  - **800 Mbit/s write**
- **Throughput Mbit/s:**
- **See alternate 600/800 Mbit and zero**
- Data Rate: 520 Mbit/s
- **Cwnd smooth**
- **No dup Ack / send stall / timeouts**



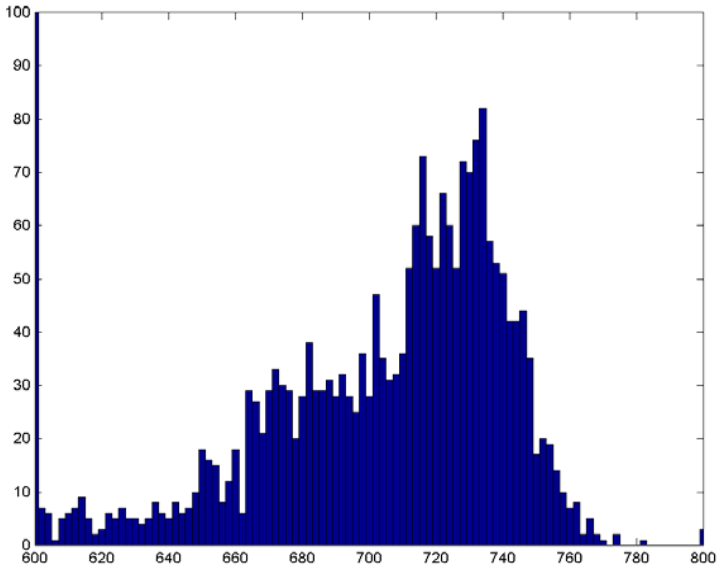


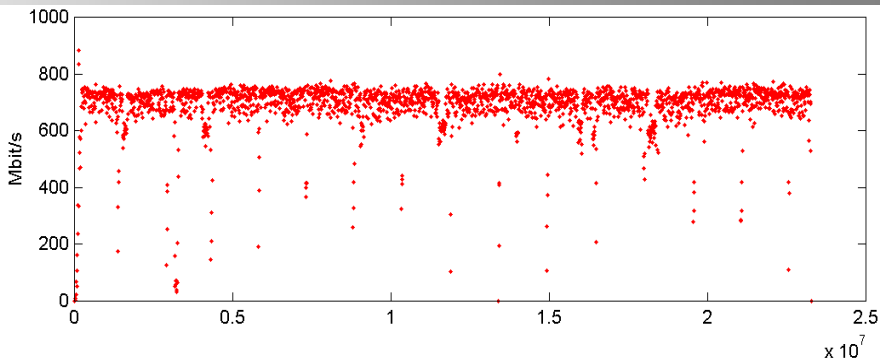


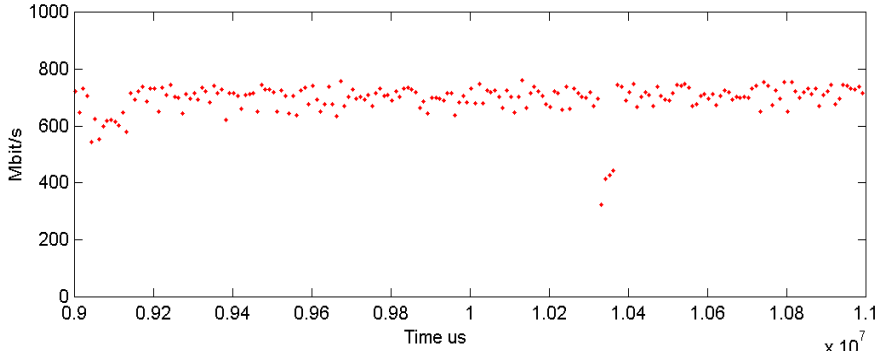


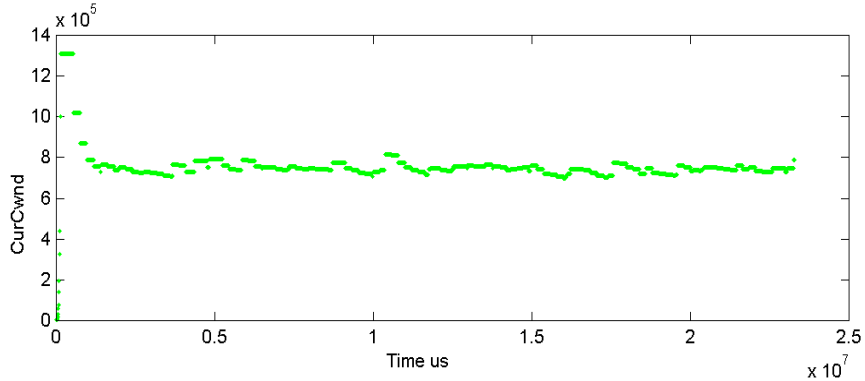
http data transfers HighSpeed TCP

- Same Hardware
- Bulk data moved by web servers
- Apachie web server out of the box!
- prototype client - curl http library
- 1Mbyte TCP buffers
- 2Gbyte file
- Throughput ~720 Mbit/s
- Cwnd - some variation
- No dup Ack / send stall / timeouts



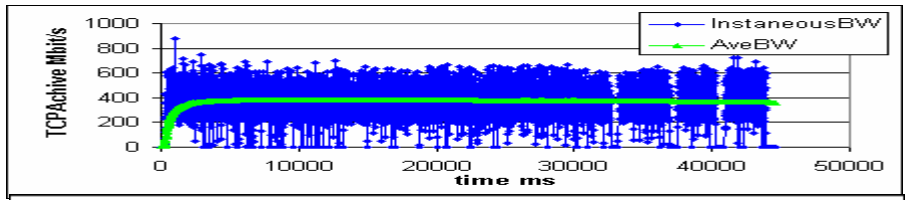


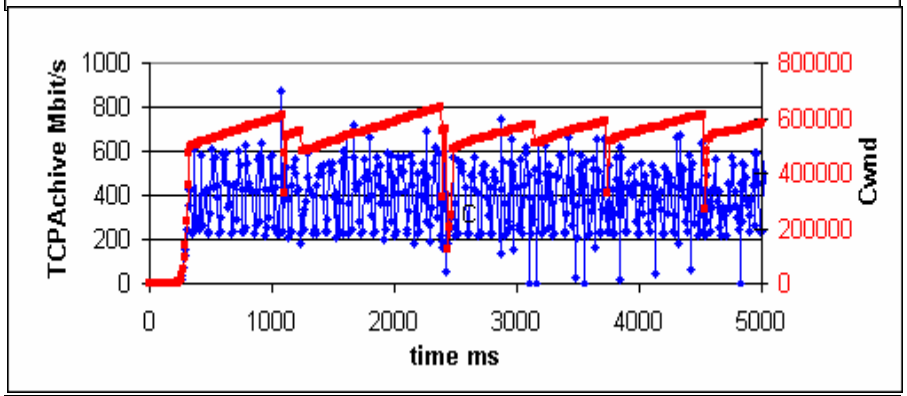




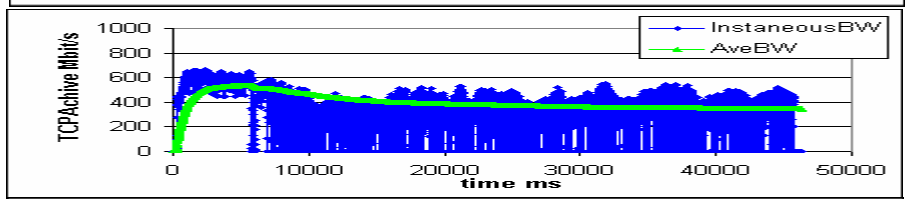
bbftp: What else is going on?

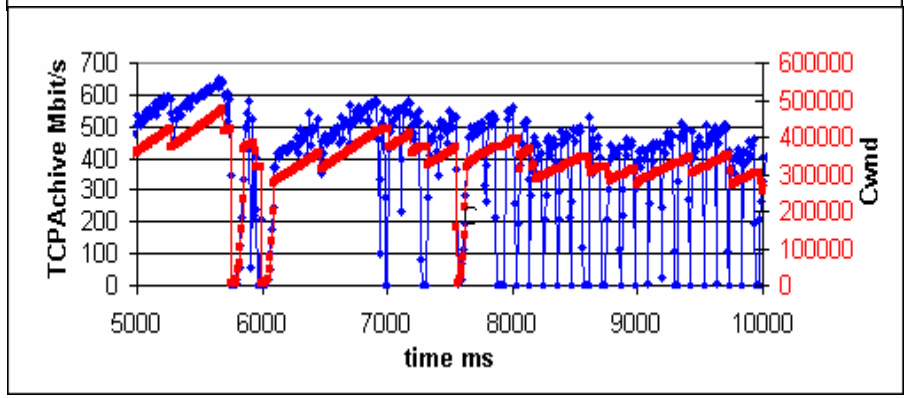
- Scalable TCP
- BaBar + SuperJANET





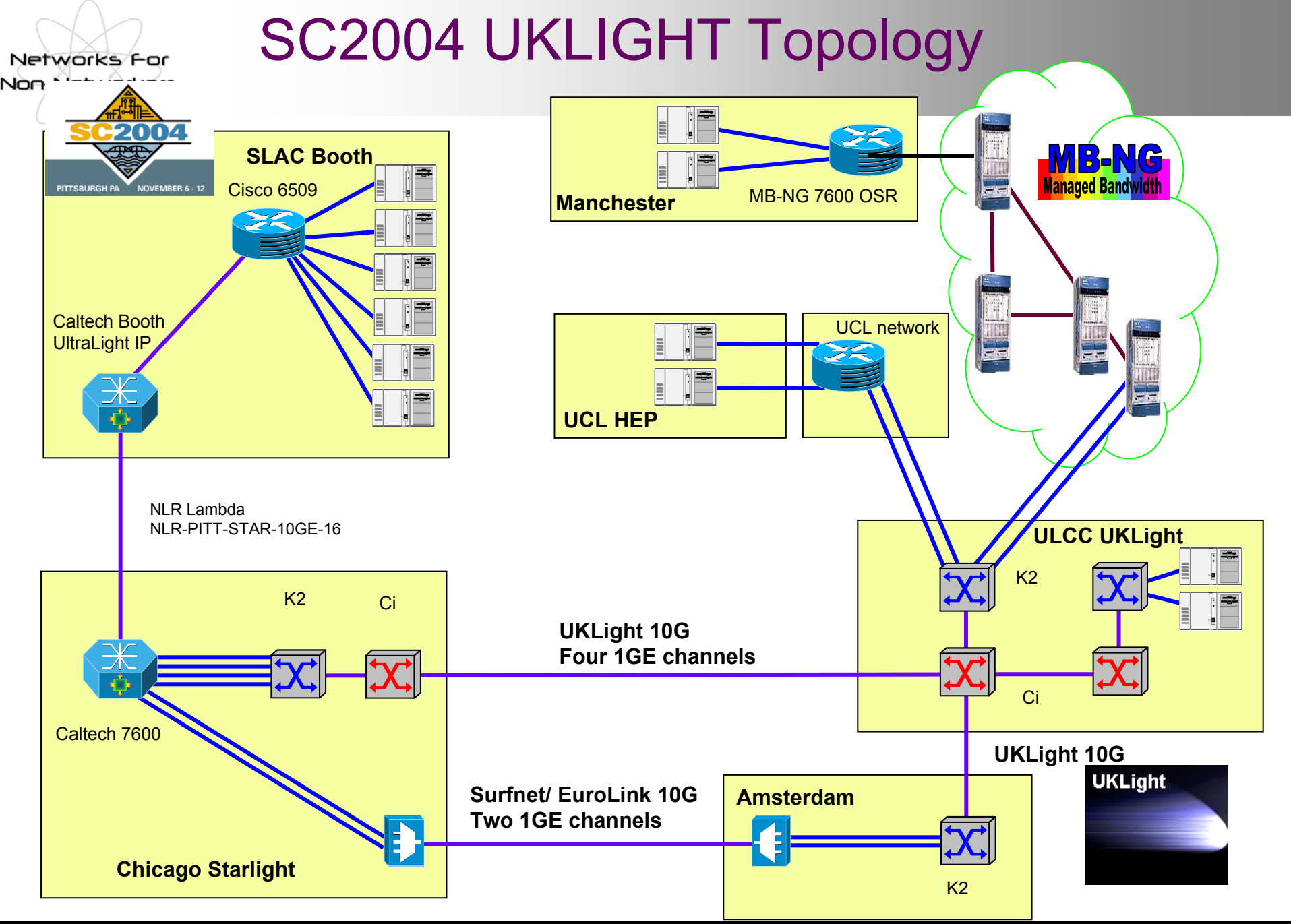
- SuperMicro + SuperJANET





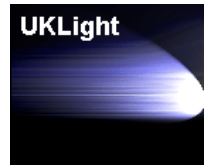
- Congestion window – duplicate ACK
- Variation not TCP related?
  - Disk speed / bus transfer
  - Application

SC2004 UKLIGHT Topology

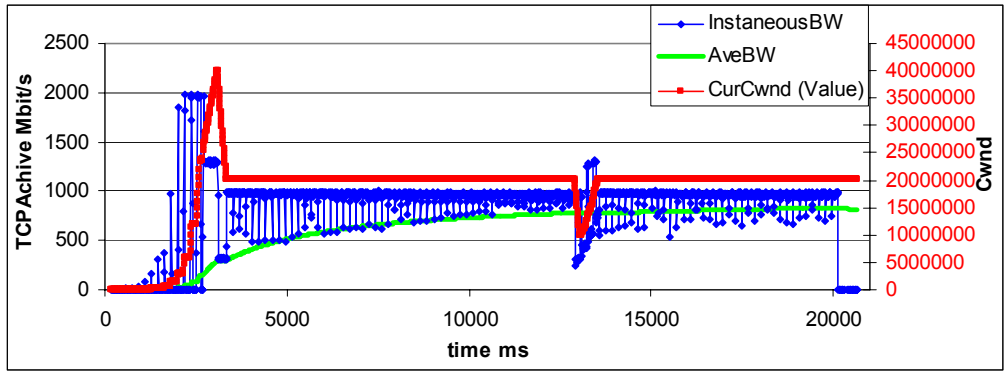


SC2004 Disk-Disk bbftp

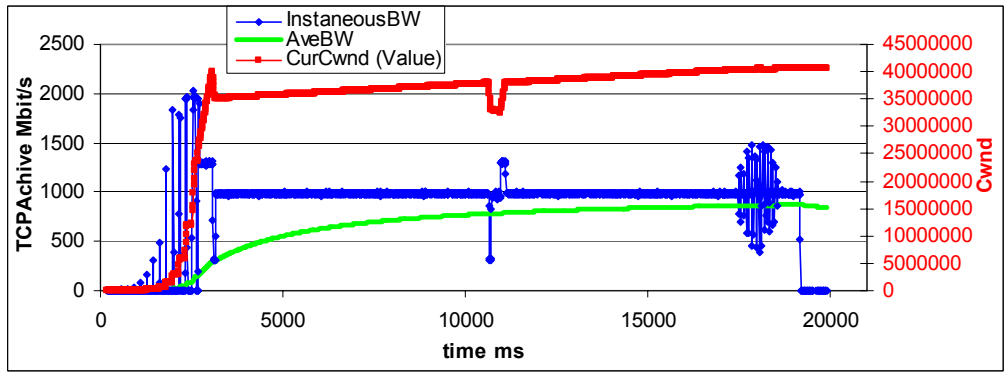
- bbftp file transfer program uses TCP/IP
- UKLight: Path:- London-Chicago-London; PCs:- Supermicro +3Ware RAID0
- MTU 1500 bytes; Socket size 22 Mbytes; rtt 177ms; SACK off
- Move a 2 Gbyte file
- Web100 plots:



- Standard TCP
- Average 825 Mbit/s
- (bbcp: 670 Mbit/s)



- Scalable TCP
- Average 875 Mbit/s
- (bbcp: 701 Mbit/s ~4.5s of overhead)

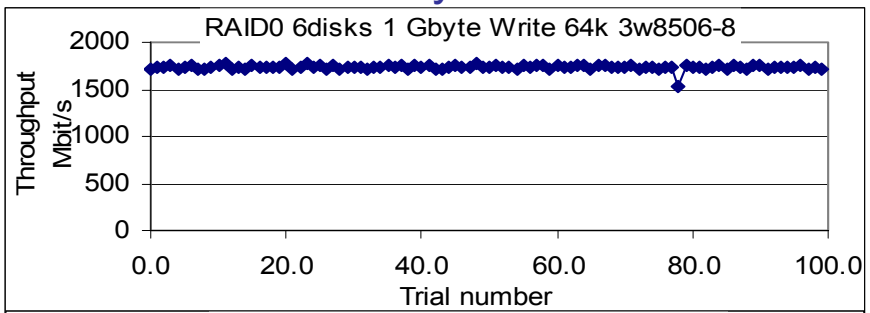


- Disk-TCP-Disk at 1Gbit/s

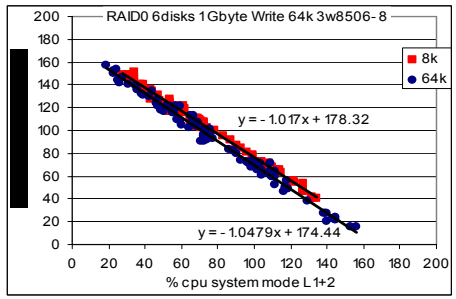
Network & Disk Interactions

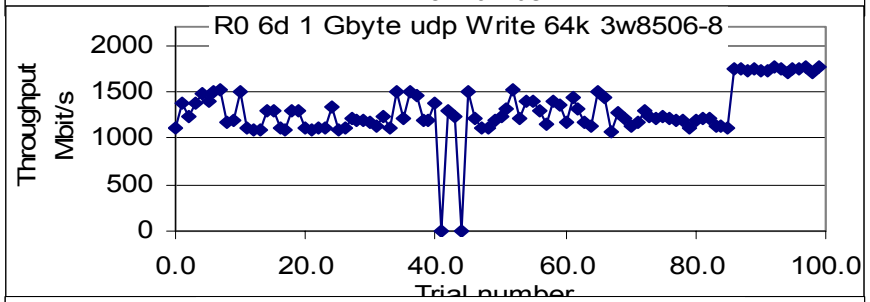
(work in progress)

- Hosts:
  - Supermicro X5DPE-G2 motherboards
  - dual 2.8 GHz Zeon CPUs with 512 k byte cache and 1 M byte memory
  - 3Ware 8506-8 controller on 133 MHz PCI-X bus configured as RAID0
  - six 74.3 GByte Western Digital Raptor WD740 SATA disks 64k byte stripe size
- Measure memory to RAID0 transfer rates with & without UDP traffic

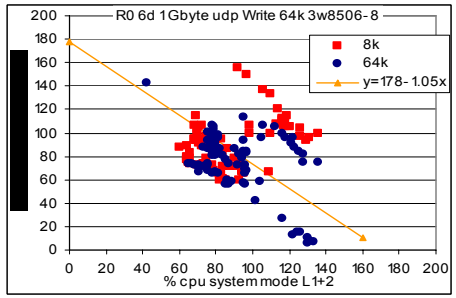


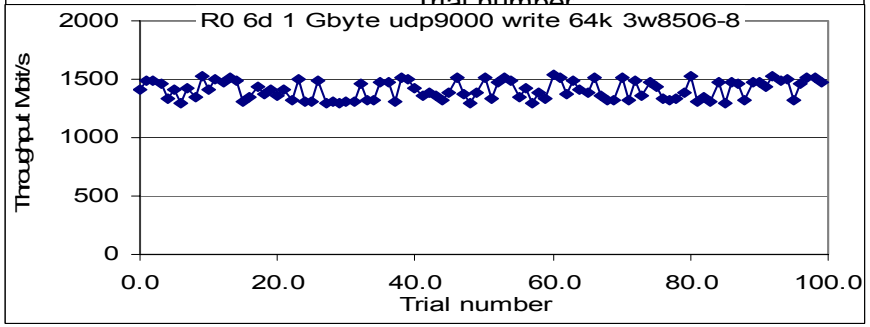
Disk write 1735 Mbit/s



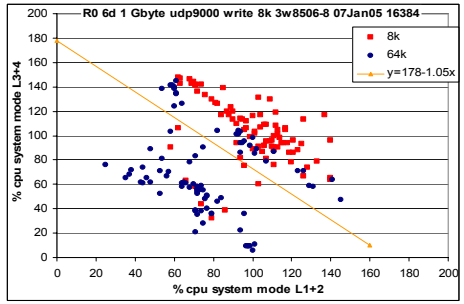


Disk write + 1500 MTU UDP 1218 Mbit/s Drop of 30%





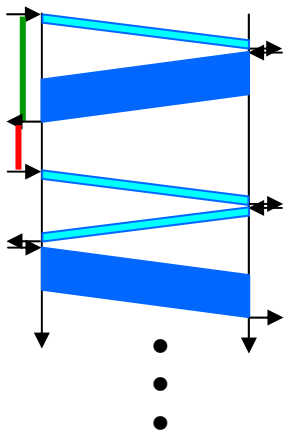
Disk write + 9000 MTU UDP 1400 Mbit/s Drop of 19%

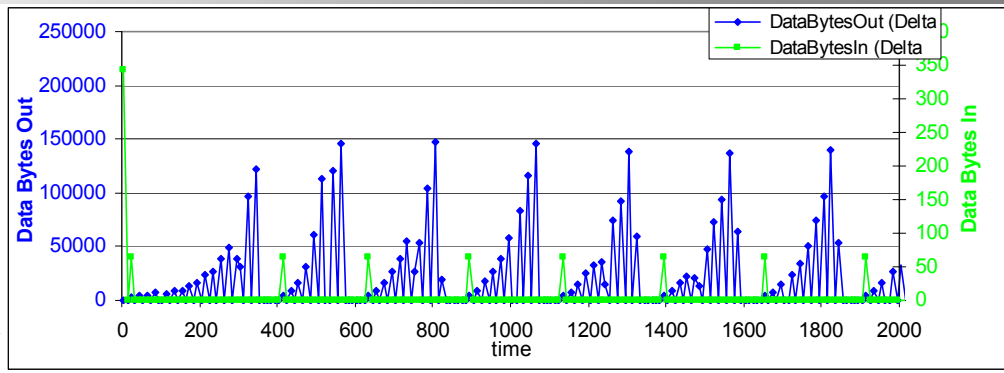


% CPU kernel mode

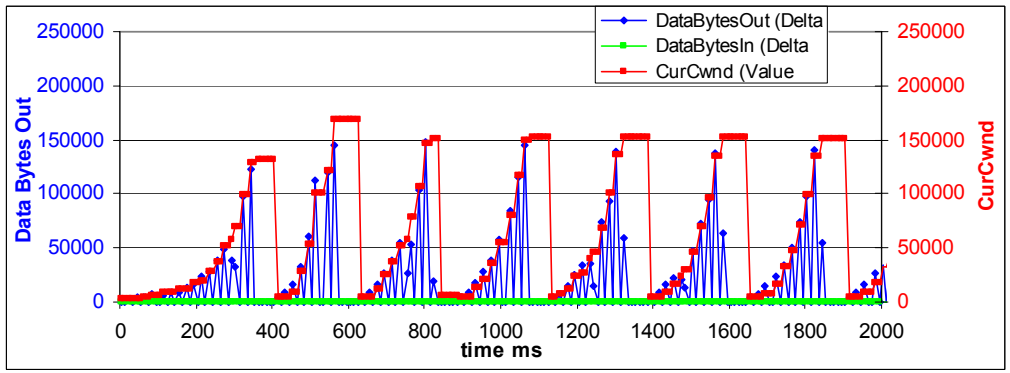
Remote Processing Farms: Manc-CERN

- Round trip time 20 ms
- 64 byte Request green
  1 Mbyte Response blue
- TCP in slow start
- 1st event takes
  19 rtt or ~ 380 ms

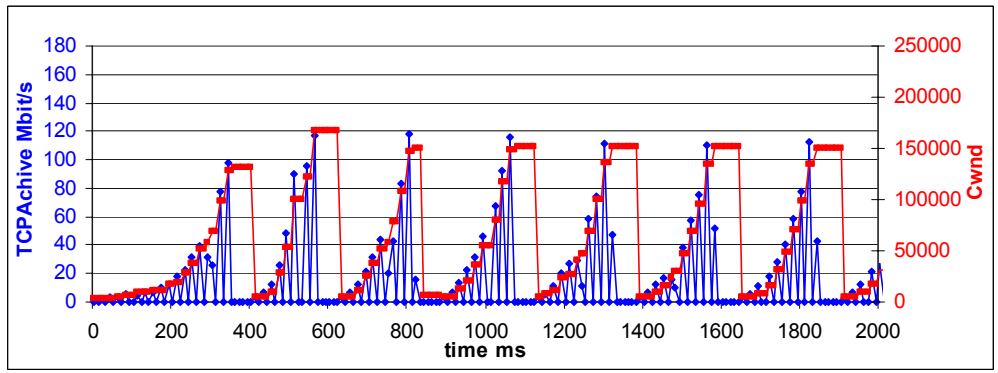




- TCP Congestion window
  gets re-set on each Request
- TCP stack **implementation**
  detail to reduce Cwnd after inactivity
- Even after 10s, each response
  takes 13 rtt or ~260 ms

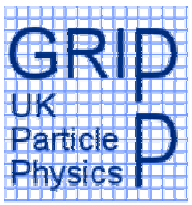


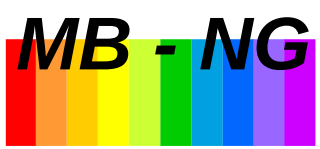
- Transfer achievable throughput
  120 Mbit/s

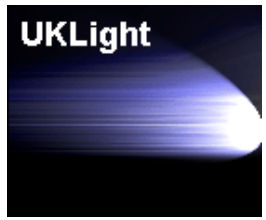


Summary, Conclusions & Thanks

- **Host is critical:** Motherboards NICs, RAID controllers and Disks matter
  - The NICs should be well designed:
    - NIC should use 64 bit 133 MHz PCI-X (66 MHz PCI can be OK)
    - NIC/drivers: CSR access / Clean buffer management / Good interrupt handling
  - Worry about the CPU-Memory bandwidth as well as the PCI bandwidth
    - Data crosses the memory bus at least 3 times
  - Separate the data transfers – use motherboards with multiple 64 bit PCI-X buses
- Test Disk Systems with representative Load
  - Choose a modern high throughput RAID controller
    - Consider SW RAID0 of RAID5 HW controllers
- Need plenty of CPU power for sustained 1 Gbit/s transfers and disk access
- Packet loss is a killer
  - Check on campus links & equipment, and access links to backbones
- New stacks are stable give better response & performance
  - Still need to set the tcp buffer sizes & other kernel settings e.g. window-scale
- Application architecture & implementation is important
- Interaction between HW, protocol processing, and disk sub-system complex






More Information Some URLs

- UKLight web site: http://www.uklight.ac.uk
- DataTAG project web site: http://www.datatag.org/
- UDPmon / TCPmon kit + writeup: http://www.hep.man.ac.uk/~rich/ (Software & Tools)
- Motherboard and NIC Tests: http://www.hep.man.ac.uk/~rich/net/nic/GigEth_tests_Boston.ppt & http://datatag.web.cern.ch/datatag/pfldnet2003/
  "Performance of 1 and 10 Gigabit Ethernet Cards with Server Quality Motherboards" FGCS Special issue 2004
  http:// www.hep.man.ac.uk/~rich/ (Publications)
- TCP tuning information may be found at: http://www.ncne.nlanr.net/documentation/faq/performance.html & http://www.psc.edu/networking/perf_tune.html
- TCP stack comparisons: "Evaluation of Advanced TCP Stacks on Fast Long-Distance Production Networks" Journal of Grid Computing 2004 http:// www.hep.man.ac.uk/~rich/ (Publications)
- PFLDnet http://www.ens-lyon.fr/LIP/RESO/pfldnet2005/
- Dante PERT http://www.geant2.net/server/show/nav.00d00h002
- Real-Time Remote Farm site http://csr.phys.ualberta.ca/real-time
- Disk information http://www.storagereview.com/

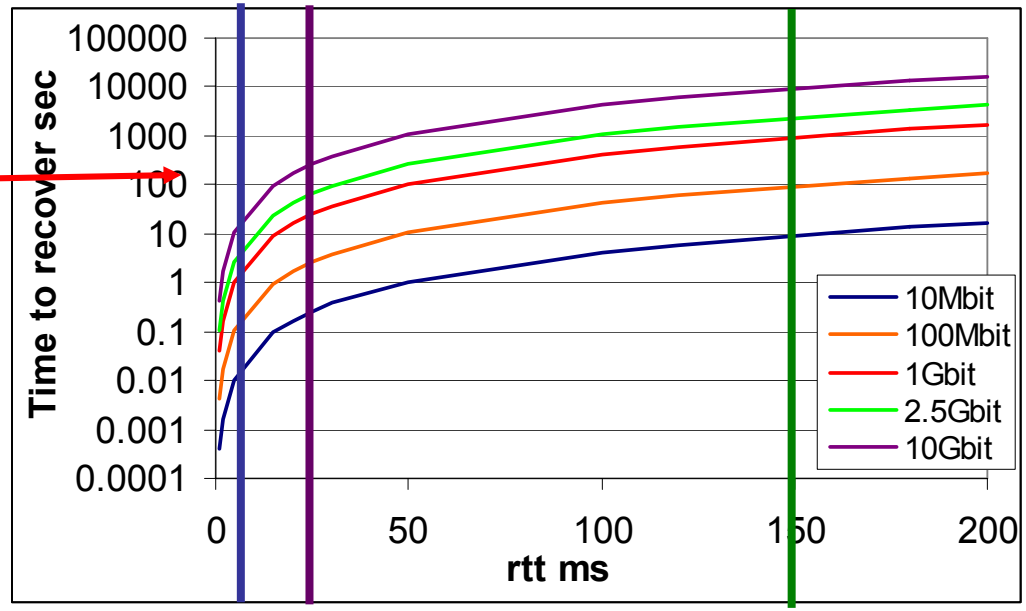
Any Questions?

Backup Slides

TCP (Reno) – Details

- Time for TCP to recover its throughput from 1 lost packet given by:

$$\rho = \frac{C * RTT^{2}}{2 * MSS}$$



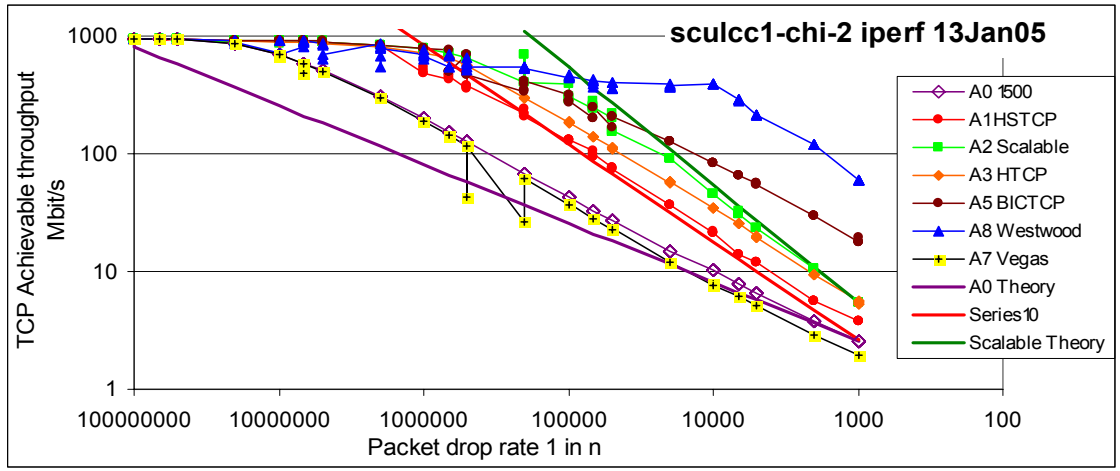
- for rtt of ~200 ms:

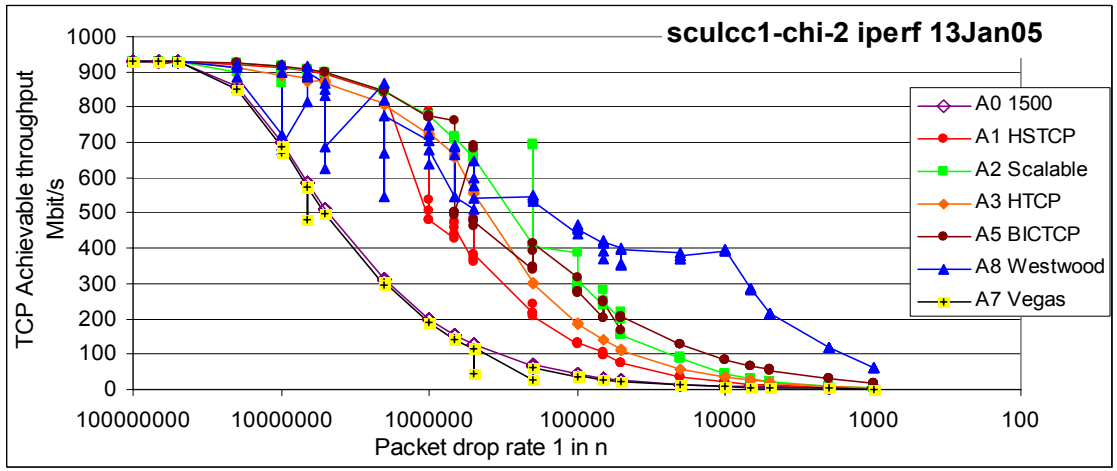
UK 6 ms Europe 20 ms USA 150 ms

Throughput	Window	Loss recovery time	Supporting loss rate
10Mbps	170pkts	17s	5.4×10^{-5}
100Mbps	1700pkts	2mins 50s	5.4×10^{-7}
1Gbps	17000pkts	28mins	5.4×10^{-9}
10Gbps	170000pkts	4hrs 43mins	5.4×10^{-11}

Packet Loss and new TCP Stacks

- TCP Response Function
  - UKLight London-Chicago-London rtt 180 ms
  - 2.6.6 Kernel
  - Agreement with theory good

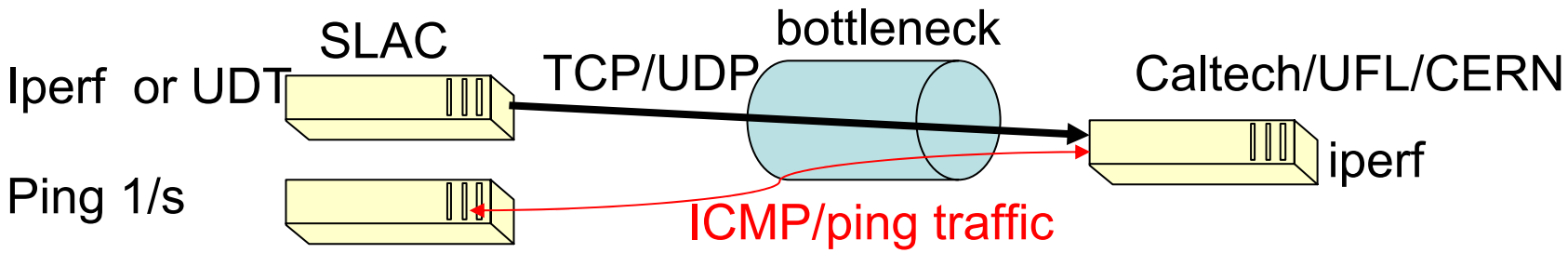




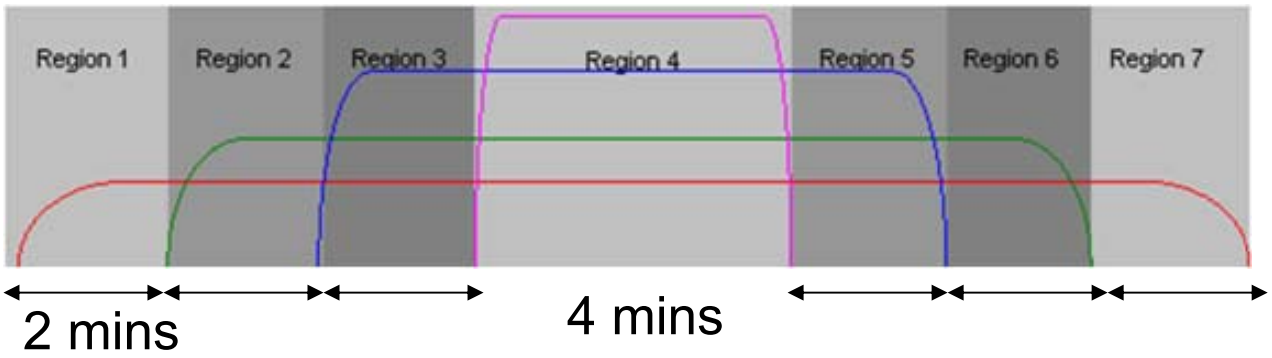
Test of TCP Sharing: Methodology (1Gbit/s)

Les Cottrell
PFLDnet
2005

- Chose 3 paths from SLAC (California)
  - Caltech (10ms), Univ Florida (80ms), CERN (180ms)
- Used iperf/TCP and UDT/UDP to generate traffic



- Each run was 16 minutes, in 7 regions



TCP Reno single stream

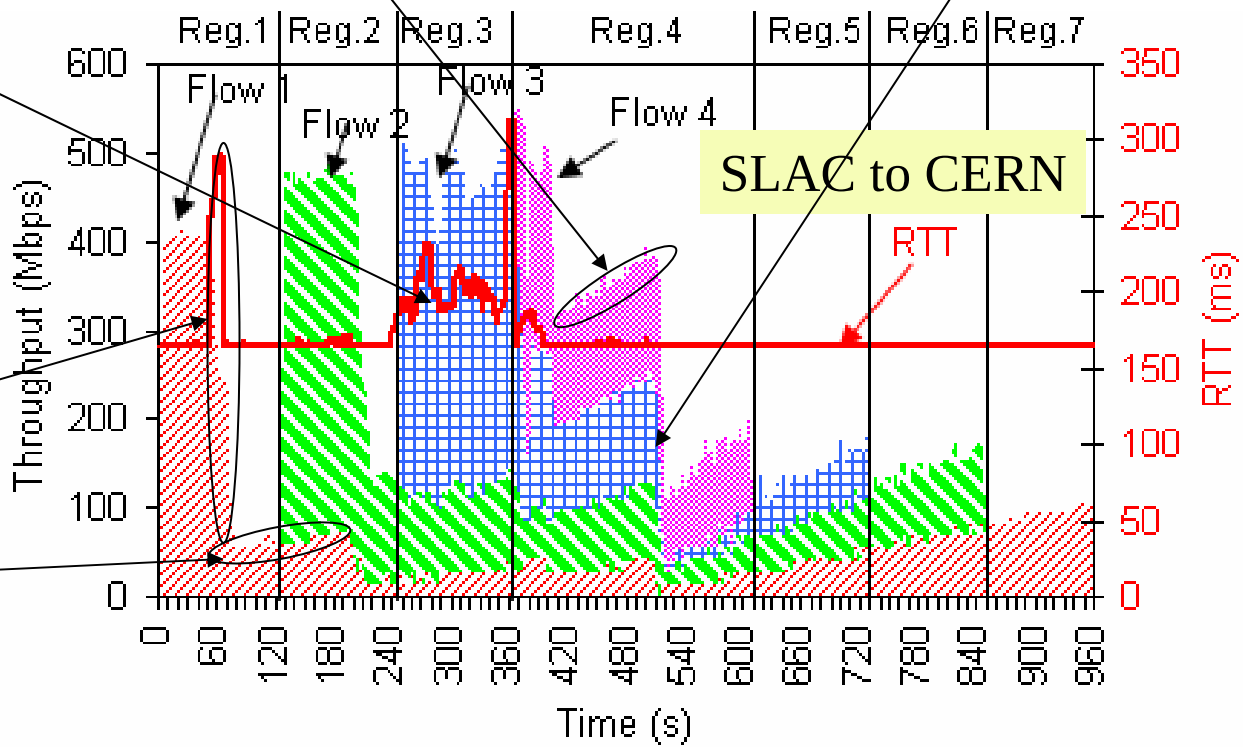
Les Cottrell
PFLDnet 2005

- Low performance on fast long distance paths
  - AIMD (add *a=1* pkt to *cwnd* / RTT, decrease *cwnd* by factor *b=0.5* in congestion)
  - Net effect: recovers slowly, does not effectively use available bandwidth, so poor throughput
  - Unequal sharing

Remaining flows do not take up slack when flow removed

Increase recovery rate

RTT increases when achieves best throughput



Congestion has a dramatic effect

Recovery is slow

Average Transfer Rates Mbit/s

App	TCP Stack	SuperMicro on MB-NG	SuperMicro on SuperJANET4	BaBar on SuperJANET4	SC2004 on UKLight
Iperf	Standard	940	350-370	425	940
	HighSpeed	940	510	570	940
	Scalable	940	580-650	605	940
bbcp	Standard	434	290-310	290	
	HighSpeed	435	385	360	
	Scalable	432	400-430	380	
bbftp	Standard	400-410	325	320	825
	HighSpeed		370-390	380	
	Scalable	430	345-532	380	875
apache	Standard	425	260	300-360	
	HighSpeed	430	370	315	
	Scalable	428	400	317	
Gridftp	Standard	405	240		
	HighSpeed		320		
	Scalable		335		

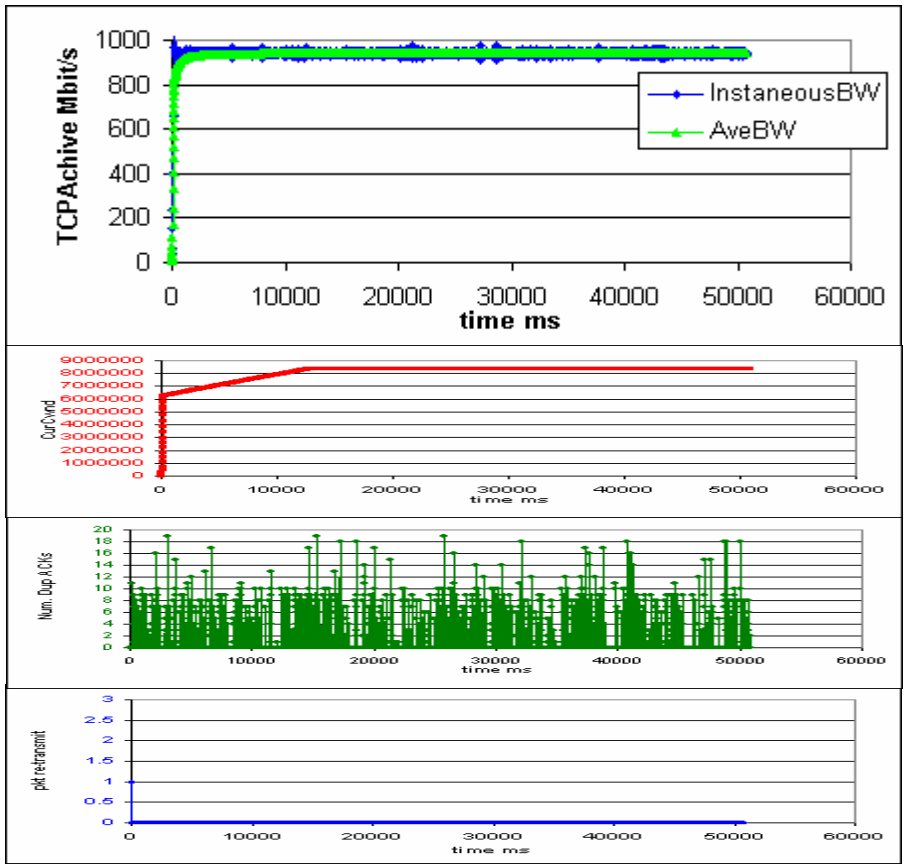
Rate decreases

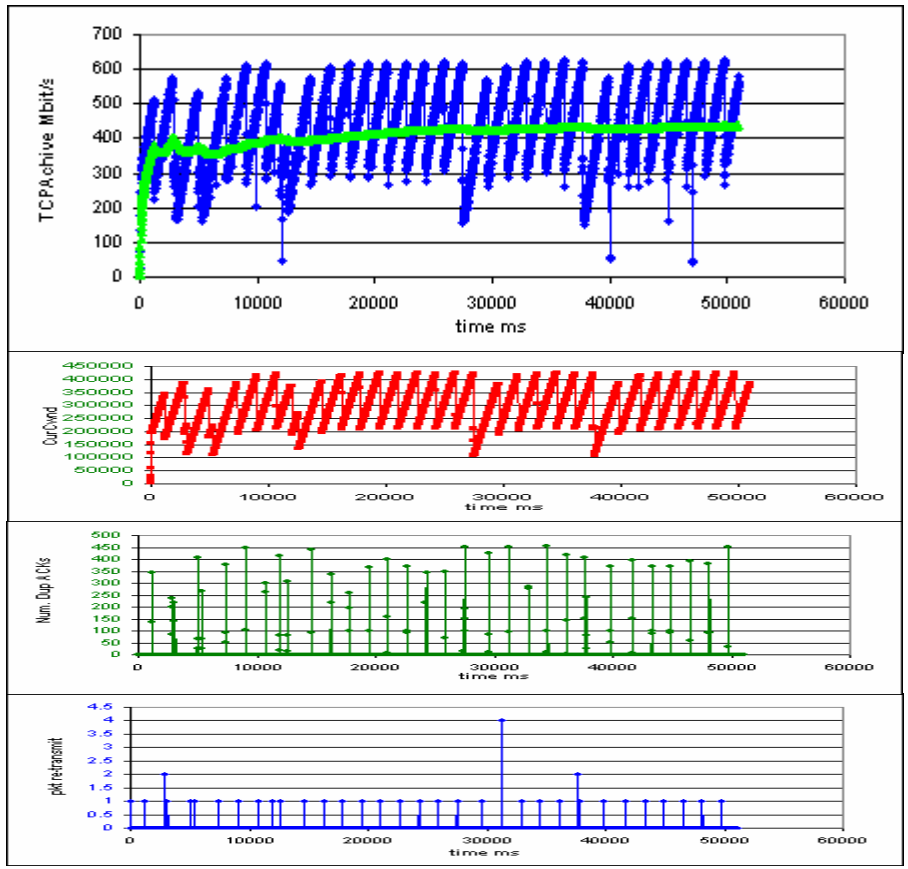
New stacks give more throughput

iperf Throughput + Web100

- SuperMicro on MB-NG network
- HighSpeed TCP
- Linespeed 940 Mbit/s
- DupACK ? <10 (expect ~400)

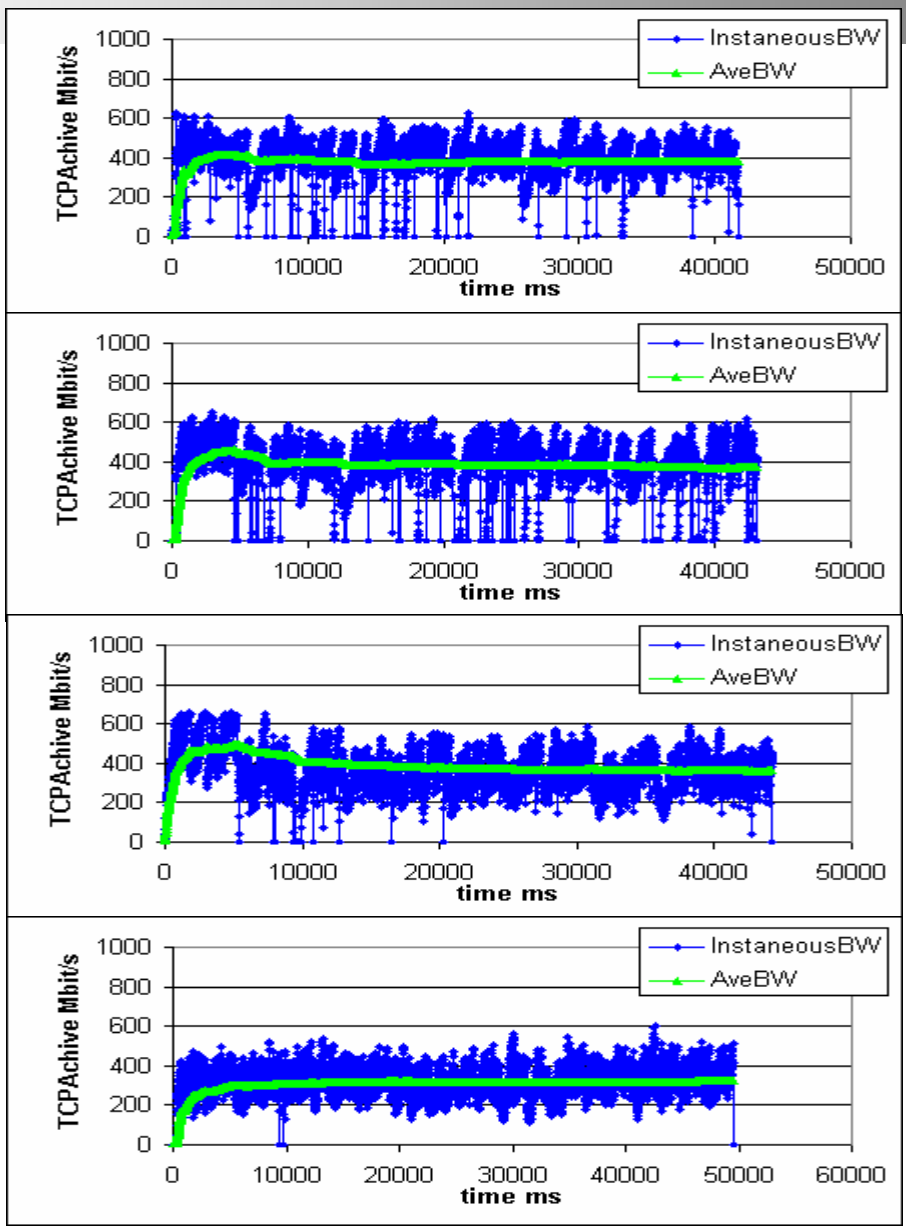
- BaBar on Production network
- Standard TCP
- 425 Mbit/s
- DupACKs 350-400 – re-transmits





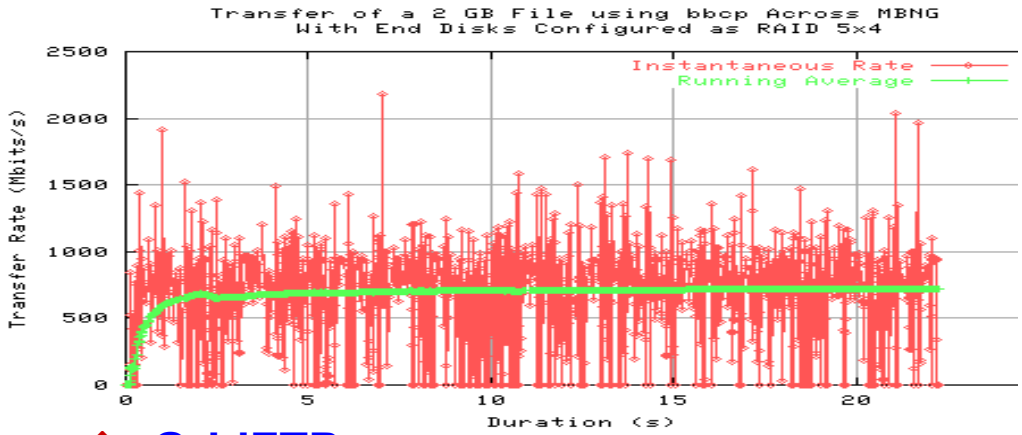
Applications: Throughput Mbit/s

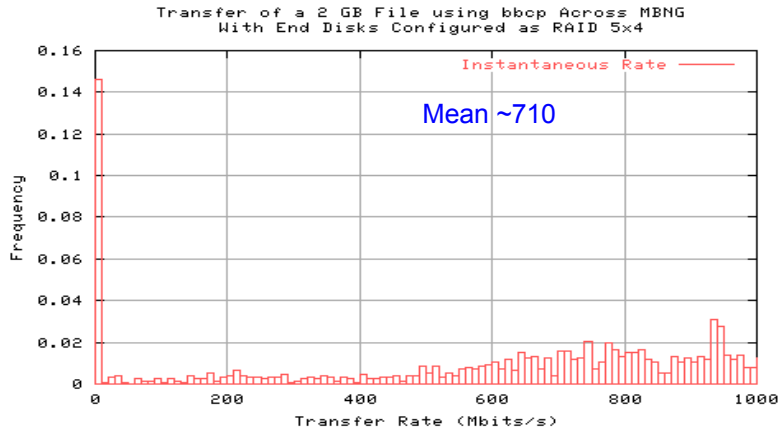
- HighSpeed TCP
- 2 GByte file RAID5
- SuperMicro + SuperJANET
- bbcp
- bbftp
- Apachie
- Gridftp
- Previous work used RAID0 (not disk limited)



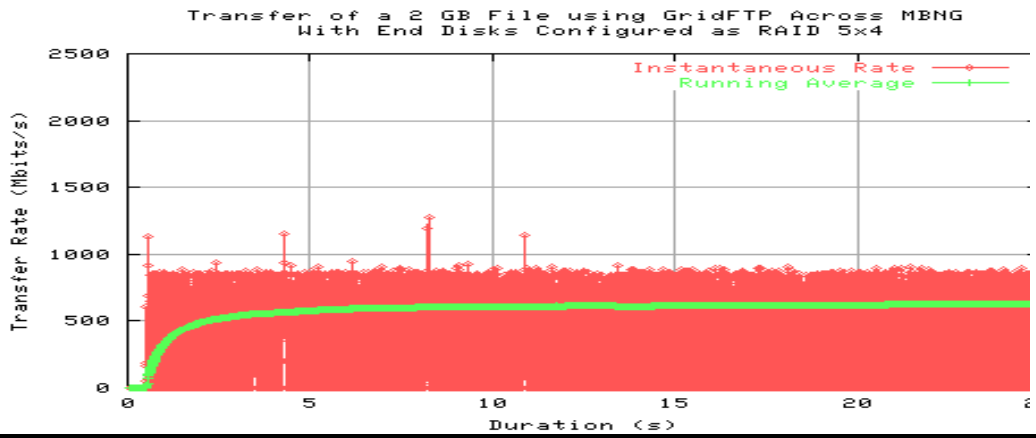
bbcp & GridFTP Throughput

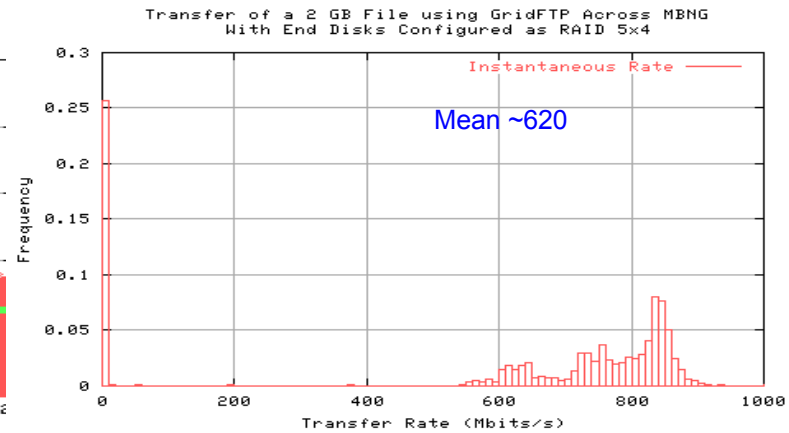
- 2Gbyte file transferred RAID5 - 4disks Manc – RAL
- bbcp
- Mean 710 Mbit/s
- DataTAG altAIMD kernel in BaBar & ATLAS





- GridFTP
- See many zeros





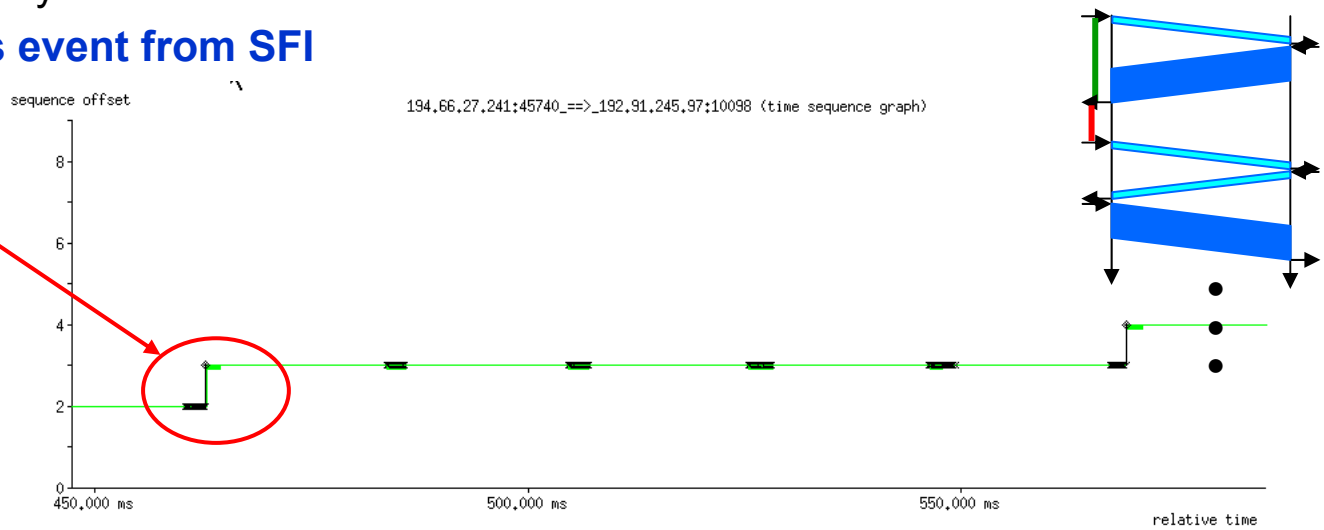
tcpdump of the T/DAQ dataflow at SFI (1)

Cern-Manchester 1.0 Mbyte event

Remote EFD requests event from SFI

Incoming event request

Followed by ACK



SFI sends event

Limited by TCP receive buffer

Time 115 ms (~4 ev/s)

When TCP ACKs arrive more data is sent.

N 1448 byte packets