Bandwidth, Throughput, IOPS, and FLOPS
The Delicate Balance

Bill Menger, Dan Wieder, Dave Glover – ConocoPhillips

March 5, 2009

Outline

The Problem – Provide enough computer to keep your customers happy (at low cost)

The Solution – Estimating needs and looking at options

The Kickoff – Never exactly works like you thought

The Tuning – Bring it into the Groove one way or the other

The Task – Get ‘er done!

The Finale – Evaluating the good, the bad, the ugly, and the beautiful

ConocoPhillips

The Problem

- We all have tasks to do in “not enough time”
- Solve your customer’s problems with an eye to the future
- Don’t box yourself into a corner (if you can help it).

ConocoPhillips

Dave's Definition of a Supercomputer

- A supercomputer is a computer that is only one order of magnitude slower than what scientists and engineers need to solve their current problems.

- This definition is timeless and assures me job security.

Dave Glover

ConocoPhillips

Solution

Task-Driven Computer Design

- Get a bunch of Geophysicists in a room, and ask:
  - What is the overall project you need to accomplish?
  - How many tasks (jobs) must be run in order to complete?
  - How long does each job take on architecture a,b,c…?
- So buy nnn of architecture xxx, back up your timeline so you can complete on time, and begin! (oh, if it were that easy)

- The FLOPS
  - We always start here – How much horsepower needed to do the job?
- The Bandwidth
  - Seems like this is the next step – MPI needs a good backbone!
- The Throughput
  - Always want a disk system to jump in with all cylinders firing.
- The IOPS/Sec
  - Never really thought about this before now. Maybe I should have!

ConocoPhillips

Task Driven Design – Case Study

- We get what we need… no more, no less
- The Need:
  - 2 iterations
  - 30,000 jobs/iteration
  - ~200 cpu-hours/job
  - 12MM cpu-hours in 50 days

ConocoPhillips

Task Driven Design – Case Study

- A solution
  - Run each job on 8 cores to get jobs down to 1 day
  - Make it easy to load/unload data for jobs with global parallel file system
  - Put enough memory into each computer to hold the job
  - Keep enough local disk to store intermediate results
  - Buy enough computers to make it happen

ConocoPhillips

Options

- Intel/AMD/Cell
- Infiniband/GigE/10GigE/Myrinet/Special
- NFS disk/Directly connected Disk (local client software)
- Specialty hardware

ConocoPhillips

Considerations

- Fit within current network infrastructure
- Think about my Reservoir Engineer and CFD customers
- What about future algorithms/approaches?

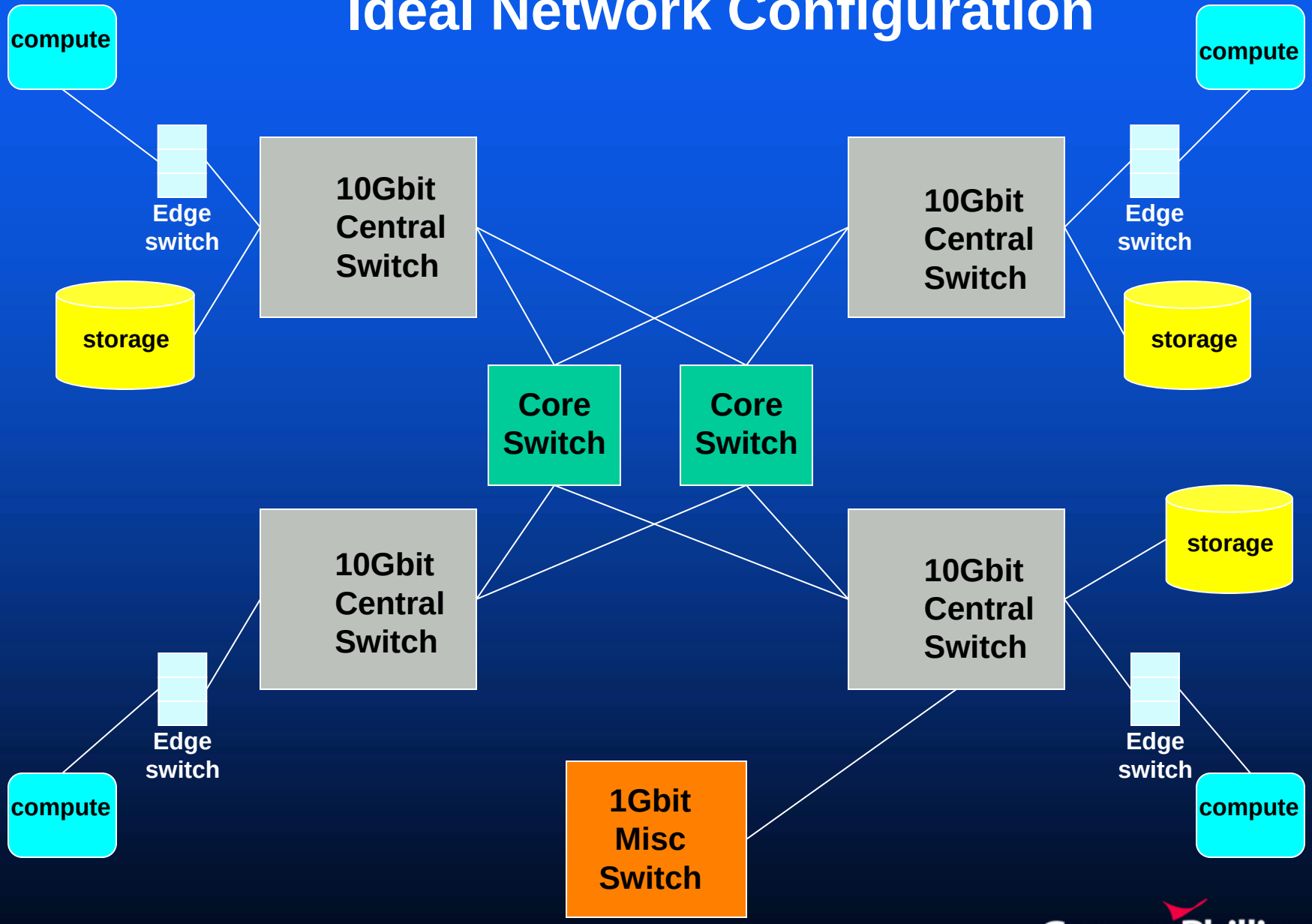
ConocoPhillips

The Chosen Design

- 1250 Dual Socket, Quad-core AMD Opterons
- 10GigE card in each node with iWarp
- Global Parallel Direct-connected disk system (client modules on each node)
- 10GigE interconnect everywhere

ConocoPhillips

Ideal Network Configuration



ConocoPhillips

The Chosen Components

- 1250 Rackable nodes, dual socket quad core AMD Opteron 2.3Ghz, 32Gbytes DRAM
- 1250 Neteffect(Intel) 10Gbit cards with iWarp low-latency software
- 42 Force-10 5210 -- 10Gbit 48 port edge switches
- 2 Foundry MLX-8 core switches (16 10 Gbit ports each)
- 4 Foundry MLX-32 core switches (128 10Gig ports each)
- 16 shelves Panasas Storage

ConocoPhillips

The Specs

- 160 Terabytes of usable storage
- 160 Gbits connectivity to storage
- 2.5Gbit full bisection bandwidth <10usec latency
- 46 Teraflops estimated capability
- 400Kwatts total power draw when running Linpack
- 24 Racks including management nodes, switches, and storage

ConocoPhillips

The Kickoff

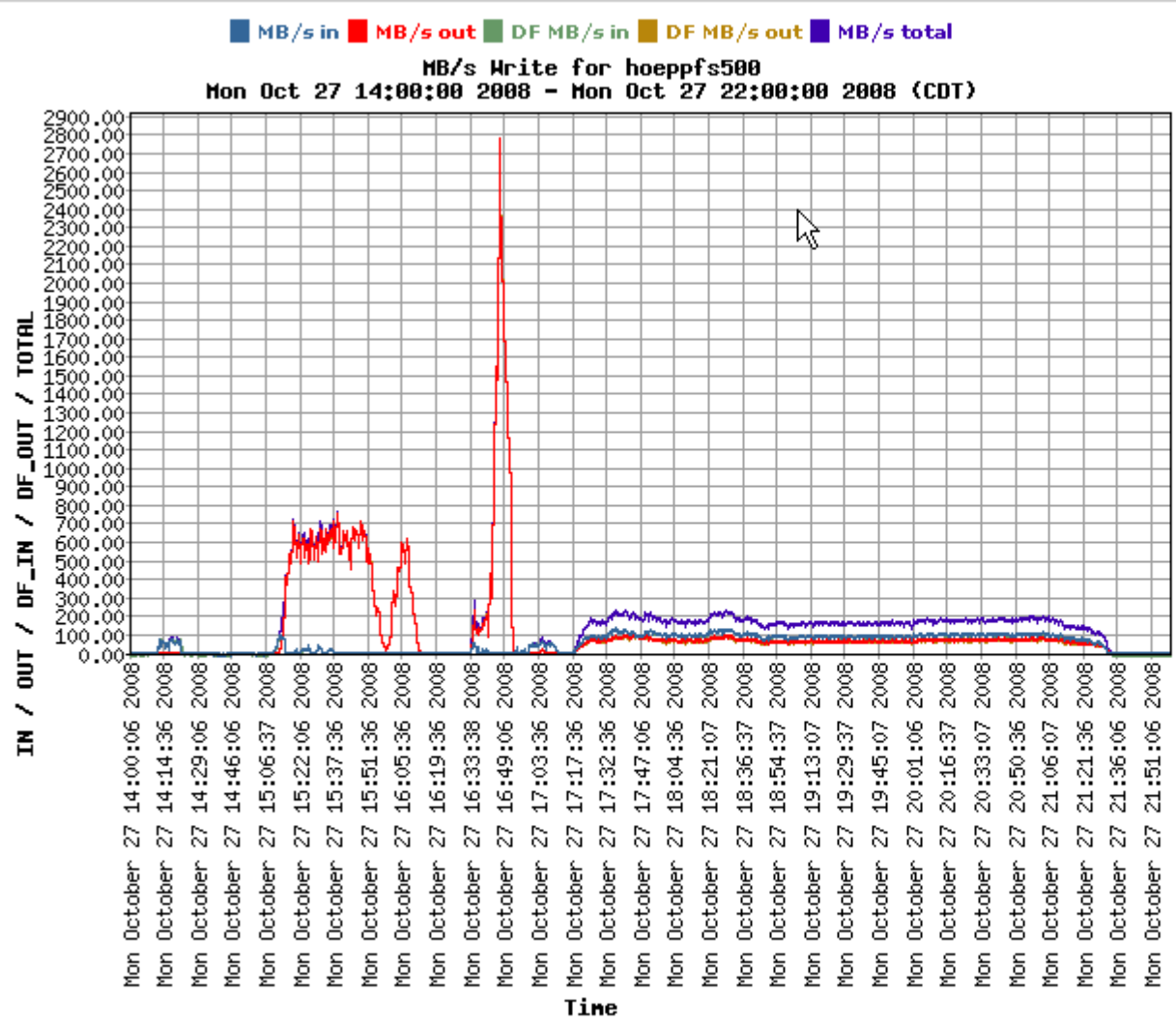
- Jobs ran (as modified) in 12 hours. Good!
- 10 Gig Network cards not available for all nodes.
- Used 1Gig to nodes, 10 Gig from edge switches to core, and 10 Gig to disks.
- 1 data space on disk (huge file system)

ConocoPhillips

A small gotcha

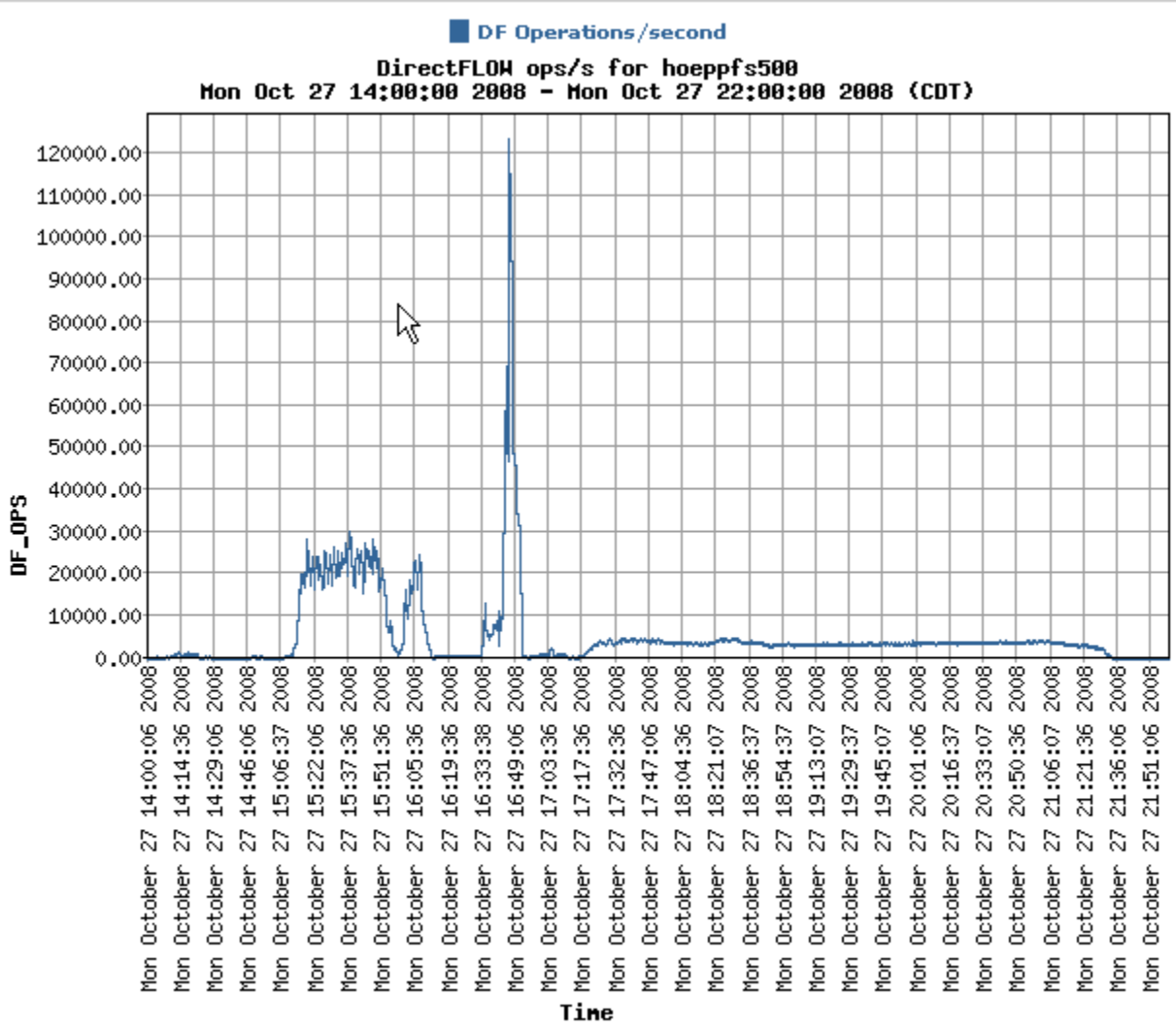
- Commercial software for processing
  - logs back to common area
  - lots of small-file initialization for each job
  - always initiates MPI so many jobs hitting user's login files all at once.
- Output file format (javaseis)
  - Nice structured sparse format, good I/O for the actual data
  - very small i/o to common area associated with file updates.

ConocoPhillips



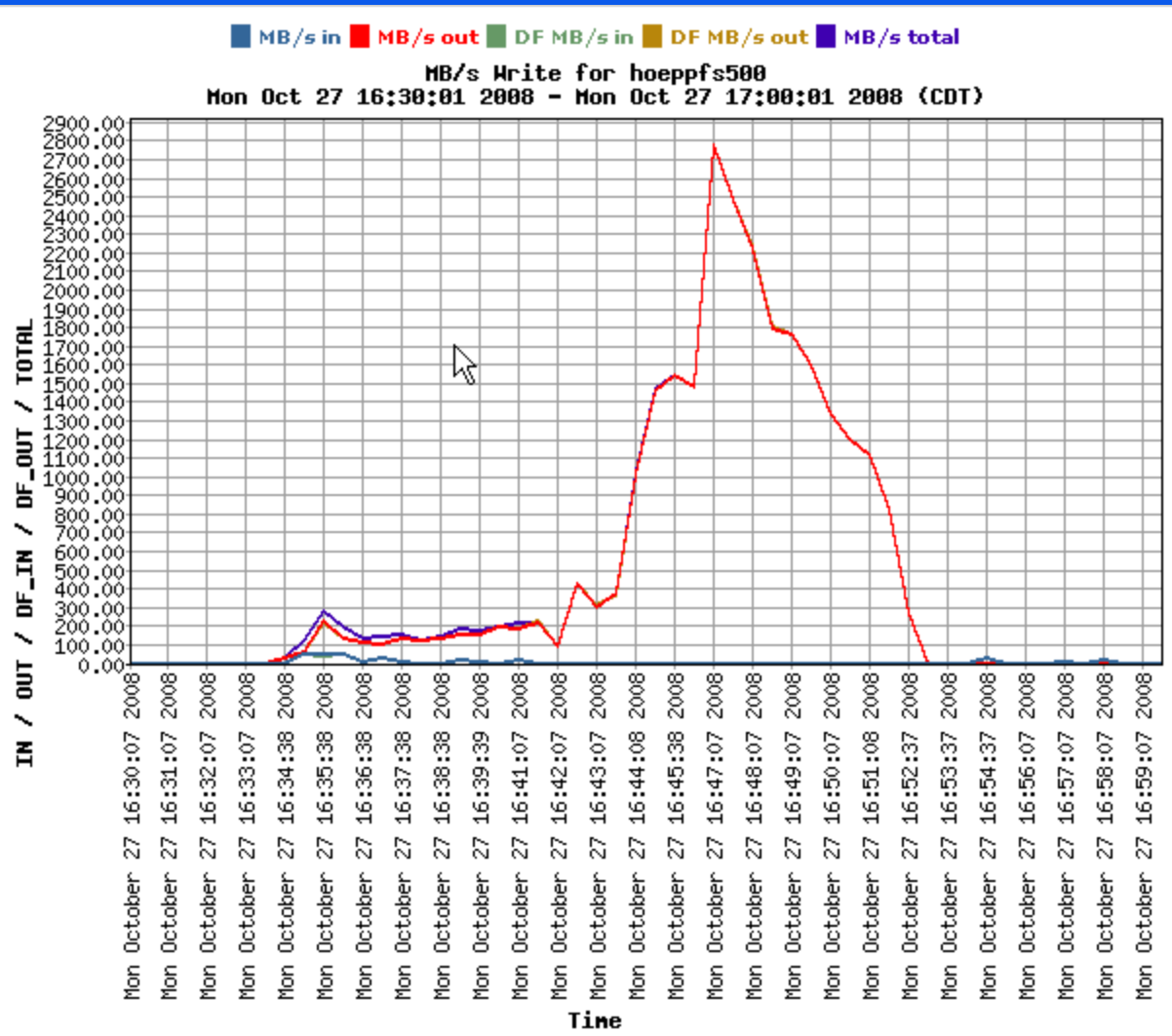
14:00 – 22:00 Disk MBytes/Sec

ConocoPhillips



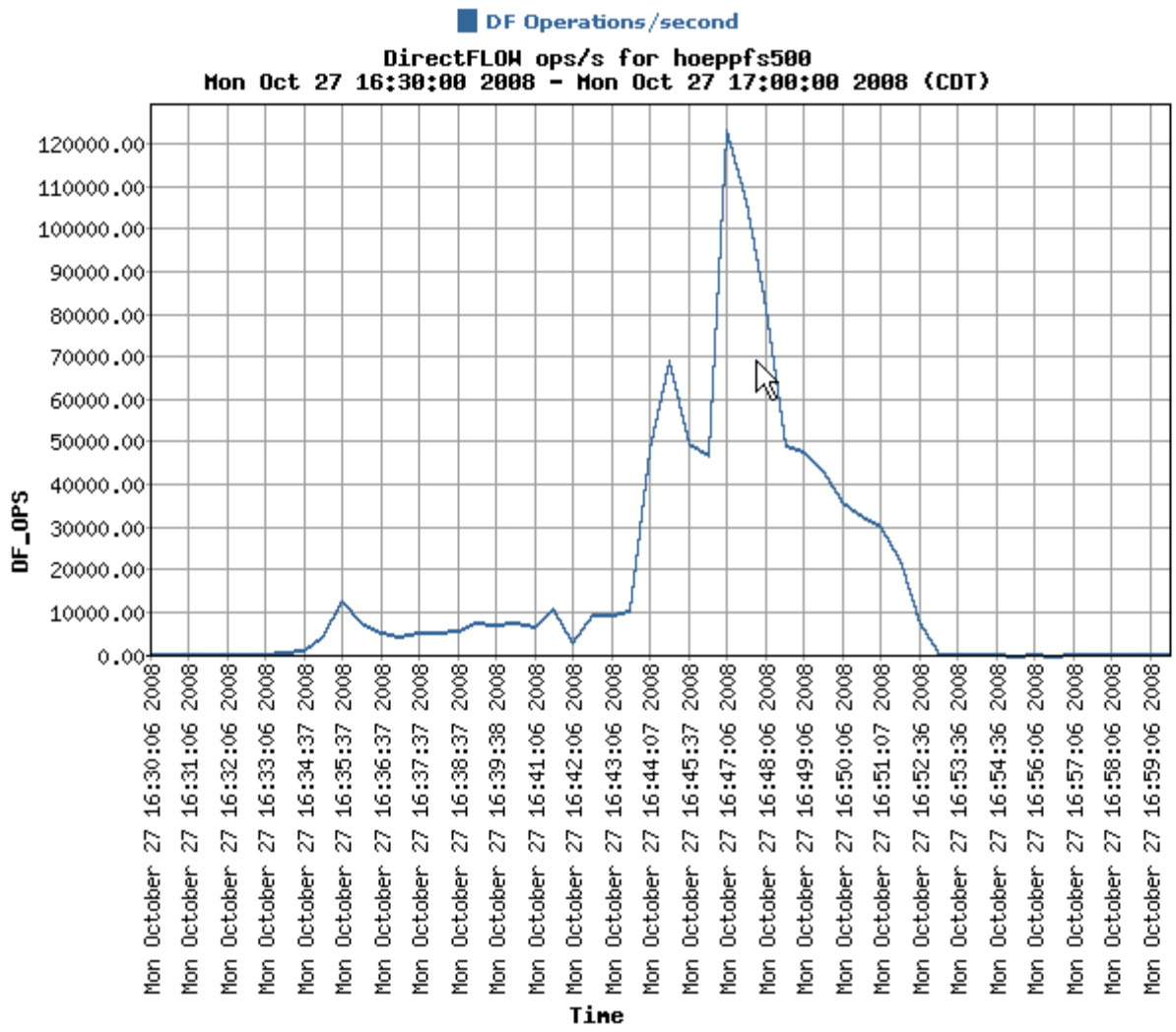
14:00 – 22:00
Disk
I/O Ops/sec

ConocoPhillips



16:30 – 17:00 Disk Mbytes/Sec

ConocoPhillips



16:30 – 17:00
Disk
I/O Ops/Sec

ConocoPhillips

Consequences

- Start up 1000 jobs in the commercial system, and watch it take 5 seconds per job to enter the queue. That's 1.4 hours!
- Jobs all reading at once, lots of i/o contention on /home and another directory
- On completion, jobs all writing small-block i/o in the common area, all at the same time. (requires read-modify-write for each operation)

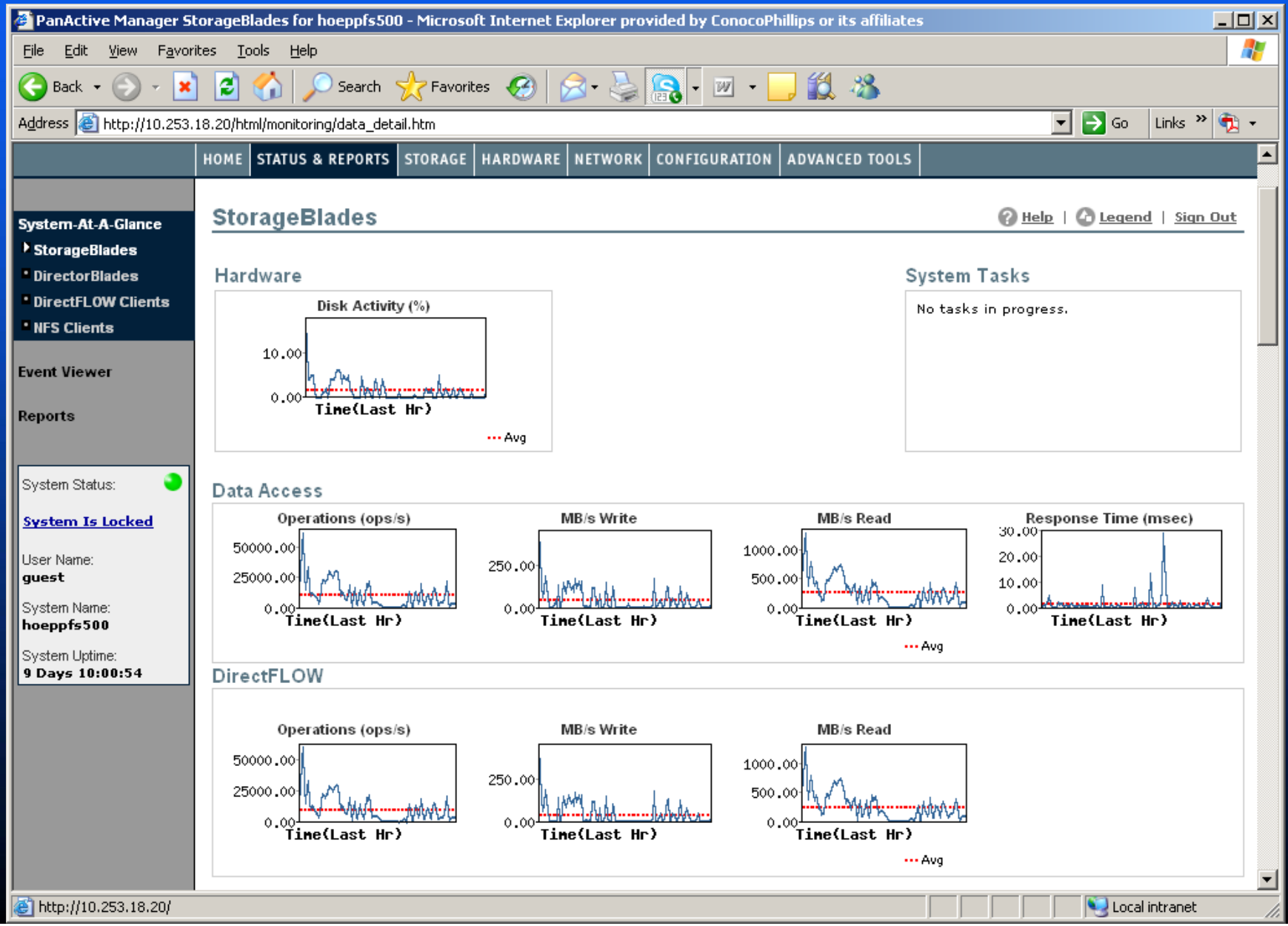
ConocoPhillips

Getting into the Groove

- First, we decided to stagger jobs. Let the auto-job builder add in a delay within each job, the delay means the jobs will start and sleep for linearly increasing times, causing a nice, steady, I/O pattern.
- Subsequent jobs are simply queued and the staggered effect persists.
- Secondly, we added a smaller data directory to house the common area. Now we have two file systems, one for large-block I/O and one for small.

ConocoPhillips

~1200 Jobs, ~50,000 IOps/Sec

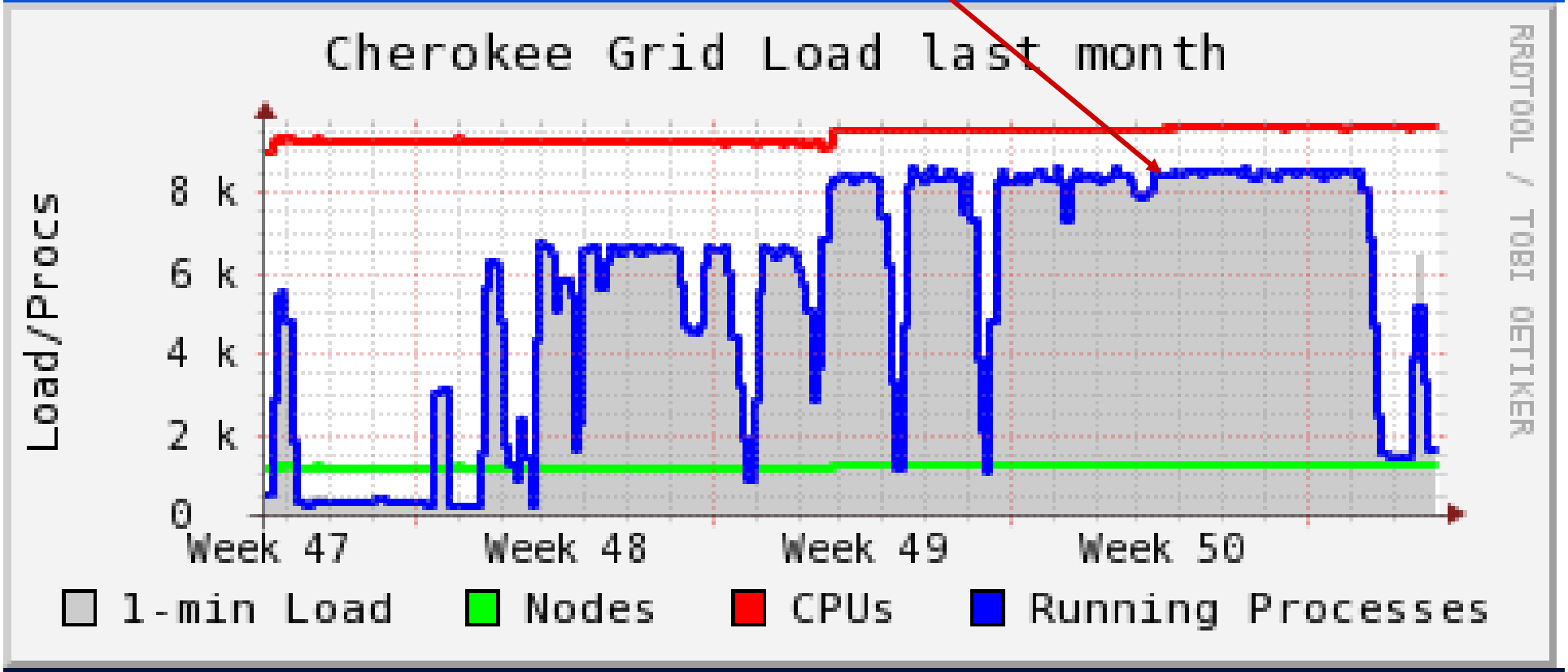


Execution

- Jobs ran well, I/O held to a maintainable level
- 1Gbit network sufficed
- Finished ahead of schedule

ConocoPhillips

Smooth Sailing!



ConocoPhillips

The good, bad, ugly, and beautiful

Good

- The job got done

Bad

- Frantic startup problems with commercial system, Disk solution

Ugly

- Realizing we undersized the IOPS

Beautiful

- System is ready for more work with “forwardly upgradable” network at the core.

ConocoPhillips

Lessons Learned

- Don’t scrimp on ANY of the following
  - Network Infrastructure (10+Gbits/s <10nsec latency)
  - I/O Bandwidth (>2500Mbytes/sec)
  - Memory (>= 4Gbytes/core)
  - I/O Ops/second (>100,000 iops)
- Vendor software is out of your control
  - (deal with it)

ConocoPhillips