NCI National Facility

- Outline
  - NCI-NF site background
  - Root on Lustre
  - Speeding up Metadata

Dr Robin Humble

Dr David Singleton

NCI

NATIONAL COMPUTATIONAL INFRASTRUCTURE

NCI National Facility

- For many years have been Australia's premier open supercomputing site (formerly “APAC-NF”)
- Current machine “vayu”
  - ~1500 nodes, ~12k Nehalem cores
  - 26 OSS's, 4 MDS's
- Wide job mix
  - Single core for 600 hour, to 8k core for 30mins
  - 128-512 core 2-3 hour jobs are “bread and butter”
- Suspend/Resume scheduling
  - 95% machine utilisation is typical
- LD_RUN_PATH aggressively set
- Root on Lustre

NCI

NATIONAL COMPUTATIONAL INFRASTRUCTURE

Root on Lustre – Images Overview

- Only 3 OS images on the whole cluster
  - Management
    - OS on local disk, rsync'd from each other
    - Holds master copy of OS images for compute & Lustre servers
    - No Lustre
  - Compute nodes, Data Movers, Login nodes
    - oneSIS with root on Lustre
    - Full OS image
  - Lustre servers
    - oneSIS ramdisk
    - Cut down 1.1G OS image

Root on Lustre – Why?

- Simplicity
  - Fewer things to fail
    - No NFS or local disks involved
- Reliability and Scalability
  - Use centralised scalable and reliable hardware
    - If Lustre is down then jobs are hung anyway. May as well put the OS there too
- Maintainability
  - One rsync from the master OS image to the OS image on Lustre updates every node immediately
  - Unlimited space for OS packages, OS variations, ...

Root on Lustre – How?

- oneSIS http://onesis.org/
  - Read-only root on NFS
  - PXE, root directory specified in dhcp, custom initramfs
  - Easy to use and modify!

- Added “lustreroot” to oneSIS kernel boot arguments. eg.
  lustreroot=mgs1@o2ib:mgs2@o2ib:/images/centos-5.5-compute-00

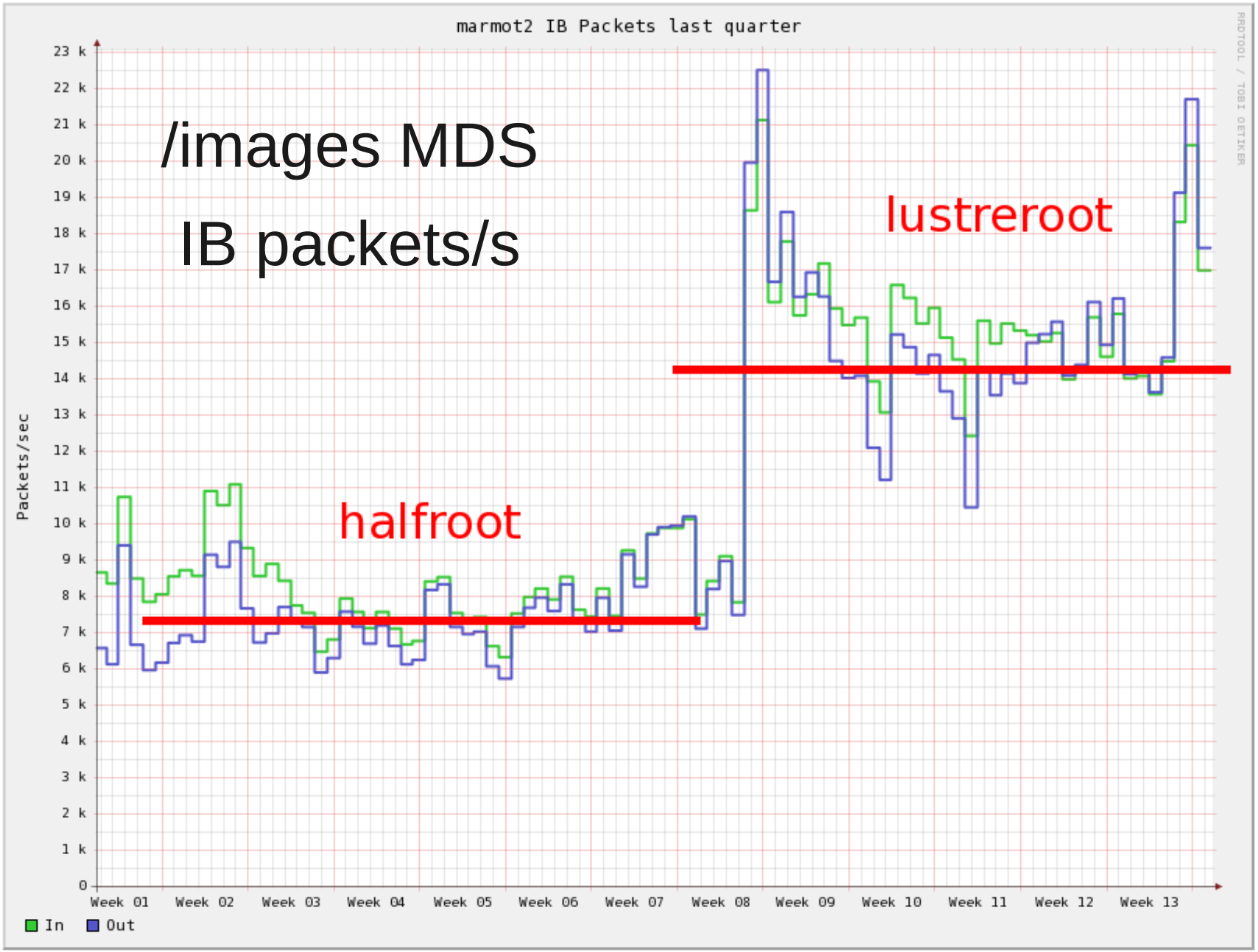
- http://nf.nci.org.au/wiki/OneSIS/Root-on-Lustre
  - 64bit busybox initramfs, glibc, mount.lustre, rsync
  - Patch oneSIS /init script
  - IB and Lustre modules in initramfs

Root on Lustre – Works Fine

Root on Lustre – Tweaks

- timeshift=12
  - Using time from BIOS can mean fs mount time is in the future
  - MDS reboot within TZ hours of node boot leads to a dead node
  - Solution is to timeshift back by >= TZ hours before mounting Lustre in the initramfs
- “halfroot”, same as “lustreroot” except
  - Have /, /lib64, /bin, /sbin in ~60MB ramfs
  - Symlinks to read-only Lustre oneSIS OS image from top level directories
  - We've been running for years in this mode, recently moved to the simpler full “lustreroot”, but might move back

Root on Lustre – Tweaks (ctd)



Root on Lustre – Pros and Cons

- Cons
  - If ptlrpcd hangs, nodes die
    - Rare deadlock in __wait_on_bit_lock due to local i/o VM pressure on our small cluster (never on the big machine)
      - Kernel bug? Lustre bug? Fundamental to network OS's?
  - Resource Management system needs to behave when the root fs (or any fs!) hangs
    - Problem exists with Root on Anything, but exacerbated by one big root fs
    - Can setup eg. pbs_mom's to use local disk or ramdisk
  - /images MDS loads up and takes a few seconds when running all-node commands
    - eg. “cexec -p uptime” takes ~12s on 1500 loaded nodes

Root on Lustre – Pros and Cons

- Pros
  - Easy to use/maintain
    - One copy of OS
    - Clients sail through Lustre server updates
  - As scalable and reliable as Lustre is
    - Performance?
      - Same as any other executable loading off Lustre
    - Will improve with Lustre (has already?)
      - Feasible to use hacks like “halfroot” or multiple /images fs's in the short term
  - Can be fully client/server cached
    - With dedicated /images OSS(s) the whole OS is trivially cacheable in OSS ram

Metadata Speed – The Problem

- Problem
  - Very slow “ls -l”
    - Uncached “ls -altrR ~” runs at ~100 files/second
    - Client-side caches help, but only when nodes aren't busy
  - Daily rsync backups taking >24hrs

Metadata Speed – Root Cause?

- MDS? No
  - Loads low
  - All fs data fits entirely in ram
    - MDT's are a 4k i/o write-only media after a while
  - With quiet OSS's, dropping all MDS caches only slows down fs sweeps by ~2x

Metadata Speed – Root Cause?

- OSS's? Yes
  - Very busy OSS's
  - Streams to read and write-through caches aggressively pushing ldiskfs inodes/dentries out of OSS ram
  - “slabtop” can see inode/dentry caches go up and down as large “ls -lR”'s try to complete. Many seeks needed to achieve this

Metadata Speed - Caching vs. Caching

- Possible Solution
  - Could turn off read_cache and writethrough_cache
    - However ~60% of i/o to our large fs gets a cache hit
    - OSS caches are clearly a win for some workloads

Metadata Speed - Caching vs. Caching

- Better (but scarier) Solution
  - Leave OSS caches on
  - Kit Westneat suggested looking at vfs_cache_pressure
    - Maybe set vfs_cache_pressure < 100 on OSS's?
- Size On Metadata feature may help too?

Metadata - vfs_cache_pressure

- What is vfs_cache_pressure?
  - Balance between pages (data) cached and inodes/dentries (ldiskfs metadata) cached
  - =100 by default
  - =0 means NEVER reclaim any inodes/dentries
    - Dangerous! Scary! Can OOM!
- But...
  - inodes are 912 bytes, dentries are 216 bytes - Tiny!
  - 1G of slab ram on 1 OSS ~= 1M files
    - Low mem OSS's shouldn't use read or write-through caches
  - inode/dentry usage grows slowly with fs

Metadata - vfs_cache_pressure=0

- Tests show clearly that any value > 0 doesn't really help, so it's all or nothing!
- Do sums, bite the bullet...
  - Set vfs_cache_pressure=0 on OSS's

Metadata - vfs_cache_pressure=0

- Result
  - 20 to 40x speedup of “ls -lR” and >10x speedup of rsync backups
    - Typical “ls -altrR ~” on an un-cached client is ~4k files/s (when client cached is 32k files/s)
    - Repeatable day to day. ie. Caches are being preserved
- Problem solved!

.... ??

Metadata – dentry problem

- Unexpected dentry problem
  - inode count goes up and down, but dentries only grow
  - dentries appear to never be deleted on OSS's, only reclaimed by VM pressure
    - Why?
  - eg. At time of writing have total across OSS's of

	Count	GByte
ldiskfs_inode_cache	27.5M	23
dentry_cache	191.7M	39
Total OSS ram		1248

Metadata - vfs_dcache_pressure

- When in doubt, hack the kernel...
  - Try a simple OSS kernel patch – add “vfs_dcache_pressure” knob that just affects the dentry shrinker

    vfs_cache_pressure=0

    vfs_dcache_pressure=100

  - But no good... makes metadata slow again

Metadata - vfs_cache_pressure=0

- Only practical solution (also the simplest!)
  - Run with vfs_cache_pressure=0
  - Occasionally set vfs_cache_pressure > 0 to reclaim dentries (and inodes)
    - Live with slow metadata speeds for a day or so until they repopulate

Summary

Root on Lustre – Summary

- Works well for us
- Easy to use/maintain
- As scalable and reliable as Lustre is
- Bit of work to setup
- See http://nf.nci.org.au/wiki/OneSIS/Root-on-Lustre

Metadata Speed - Summary

- The Problem
  - Very slow “ls -l” and daily rsync backups taking >24hrs
- The Cause
  - Very busy OSS read and write_through caches pushing ldiskfs inodes/dentries out of ram
- A Solution
  - (Carefully) set vfs_cache_pressure=0 on OSS's
  - Occasionally set to >0 to cull dentries
- Result
  - 20 to 40x speedup of “ls -lR” and >10x speedup of rsync backups

nf.nci.org.au

NCI

NATIONAL COMPUTATIONAL INFRASTRUCTURE