Introduction to Red Hat Cluster Infrastructure and Global File System

Gary Shi, SHLUG
garyshi@gmail.com

Sep, 2005

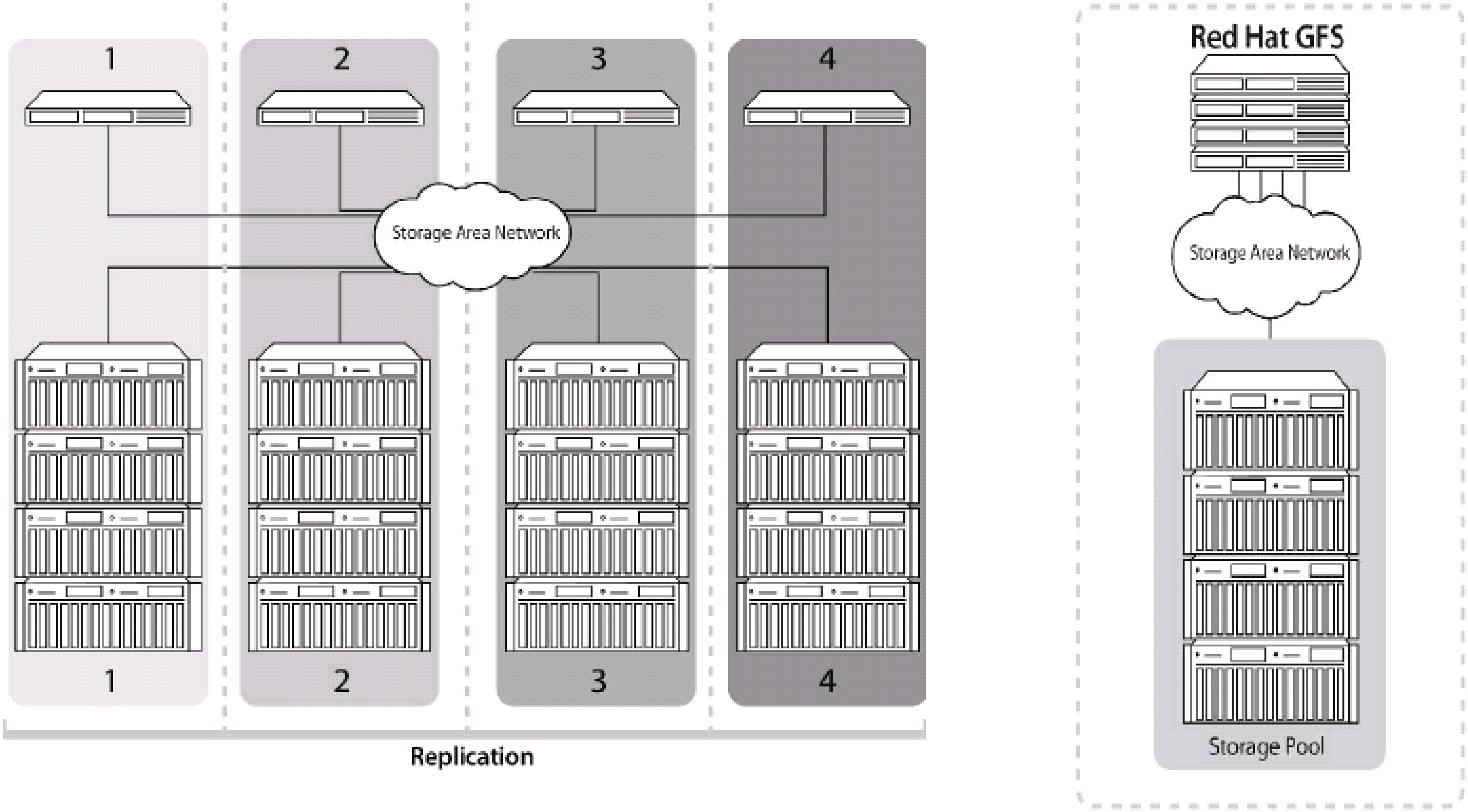
Agenda

- Overview
- Architecture Details
- Testing and Benchmark

Global File System

- Access the same file system on the same physical storage device (usually SAN)
  - Concurrent access
  - Ease of management
- Fully POSIX-compliant
- Scales up to 300+ nodes
- Multipath and Multi-Volume
- Supports x86, Itanium and AMD64/EM64T

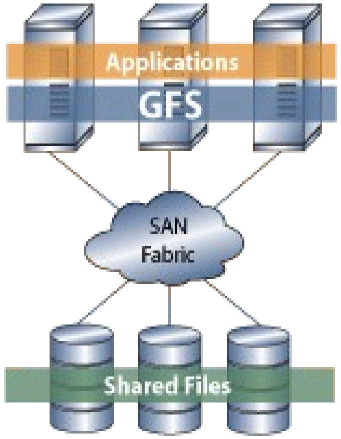
Benefit of Shared File Systems

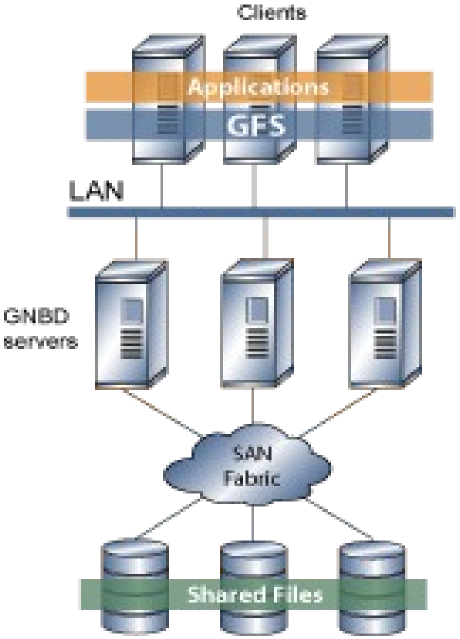


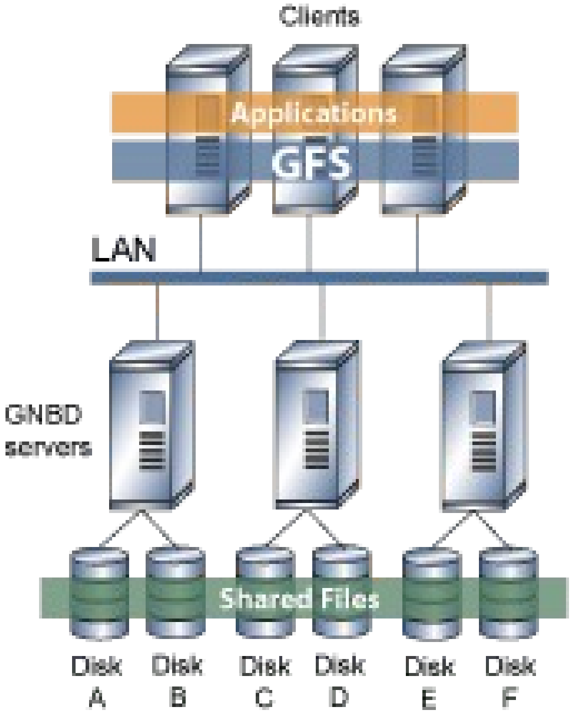
Problems with NFS

- Central server
- < 10 heavy loaded clients
- Single Point of Failure
- Broken locking

Possible Deployments







Typical Applications

- Database: Oracle RAC
- Media streaming
- Message service
- SSI clusters

Similar Products

- Oracle OCFS
  - Not POSIX compliant
- Oracle OCFS2
  - POSIX compliant
  - Based on EXT3
- IBM GPFS
  - For AIX on RS/6000
  - POSIX compliant
  - No mmap() support, and limited stat()

Agenda

- Overview
- <u>Architecture Details</u>
- Testing and Benchmark

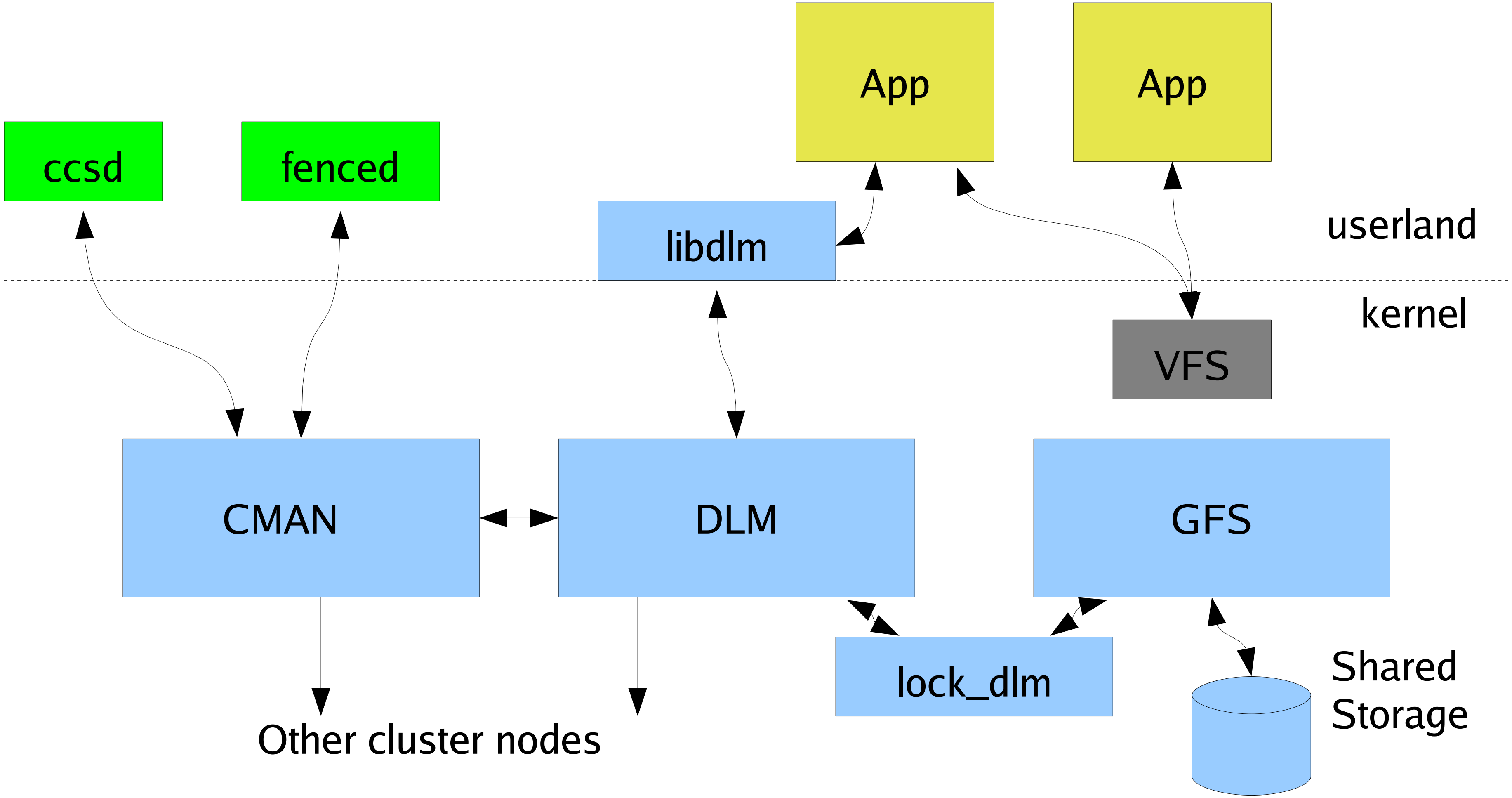
GFS History

- Originally an open source project at the University of Minnesota
- 2001, Research team formed Sistina and make GFS proprietary, open source project continues as OpenGFS
- Jan 2004, Acquired by Red Hat for $31M
- Jun 2004, Red Hat GPLed GFS

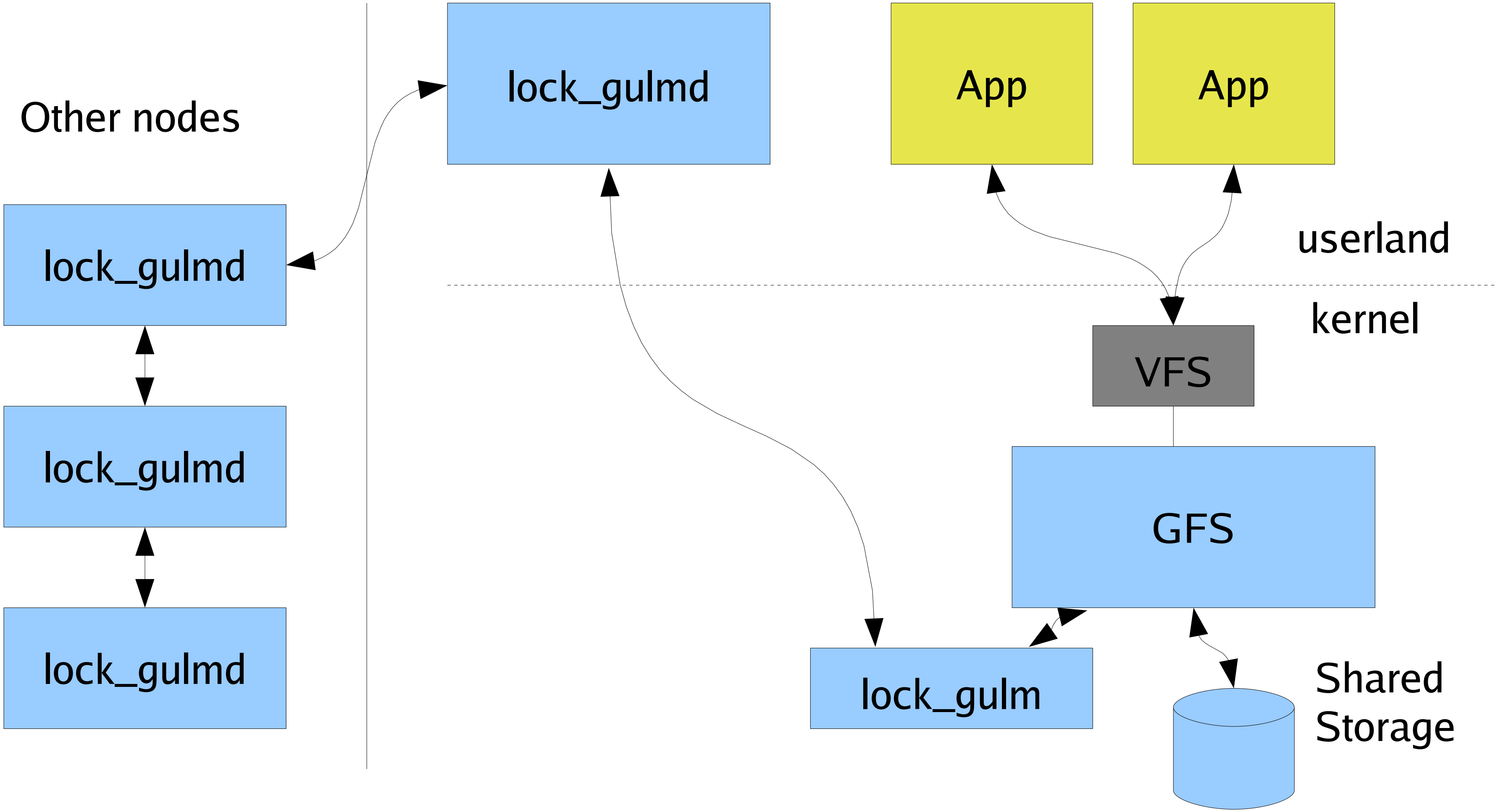
GFS Versions

- 6.0
  - 2.4 kernel, RHEL3
  - GULM (Grand Unified Lock Manager)
  - Standalone architecture
- 6.1
  - 2.6 kernel, RHEL4
  - DLM (Distributed Lock Manager)
  - Integrated into SCA (symmetry cluster architecture)

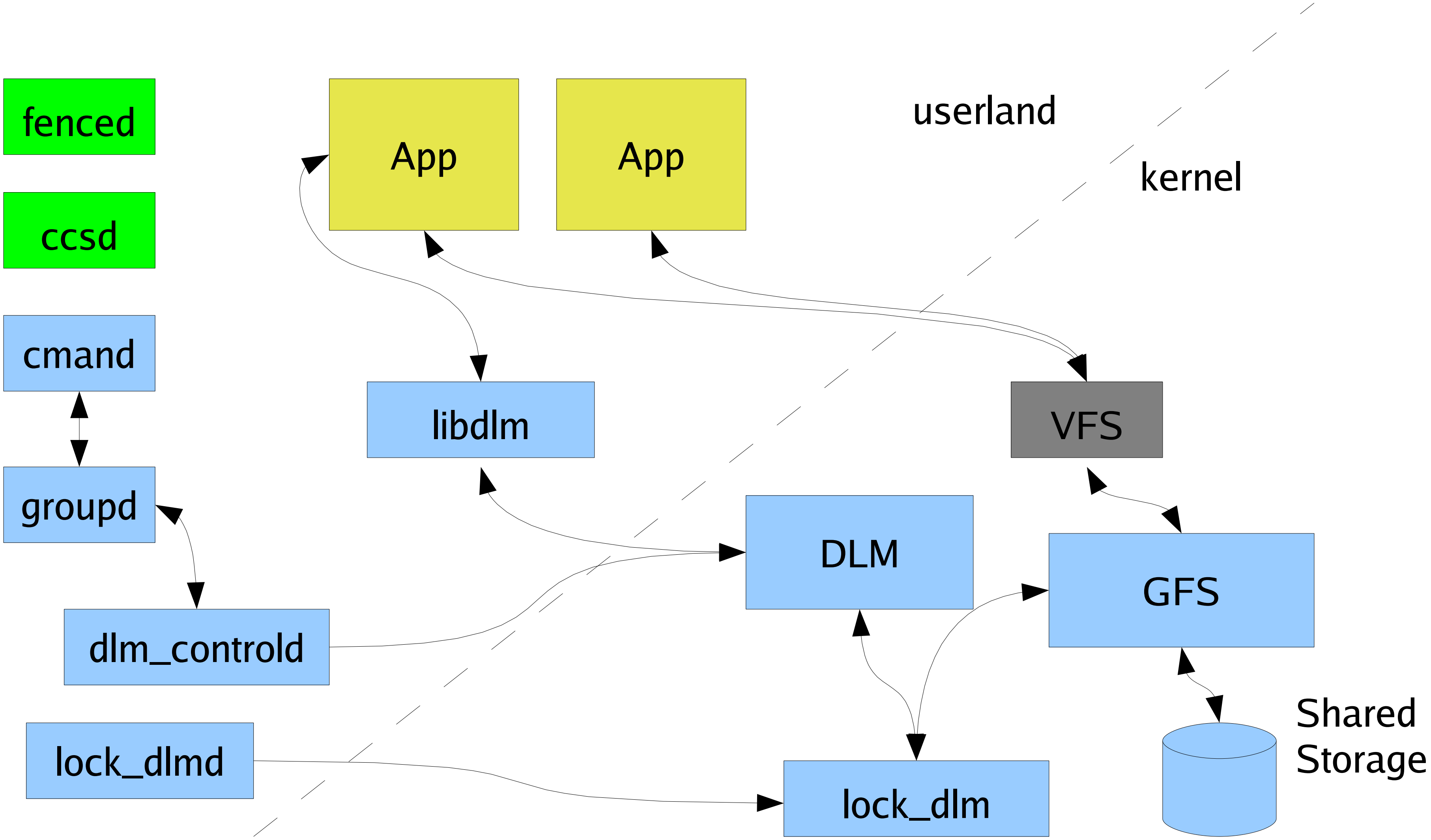
CMAN/DLM Architecture



GULM Architecture



Future Architecture



Symmetry Cluster Architecture

- Because cluster is not just GFS!
- Services can be modularized and generalized
  - Node membership
  - Fencing and fail-over
  - Distributed lock manager
- Base of new Red Hat Cluster Suite
- Base of new cluster-aware OS

Cluster Concept Model

- Cluster
  - Nodes
  - Fence Devices
  - Managed Resources
    - File systems (local, NFS, GFS, etc.)
    - Floating IP Addresses
  - Services
    - Fail-over domains

SCA Components

- CCS: Cluster Configuration System
- CMAN: Cluster Manager
- FENCE: I/O Fencing System
- GDLM: Global Distributed Lock Manager
- LOCK_GDLM: GFS Lock Module for the GDLM
- GFS: Global File System
- CLVM: Cluster Logical Volume Manager
- CSNAP: Snapshot of shared block device

Agenda

- Overview
- Architecture Details
- Testing and Benchmark

Test Environment

- Hardware
  - GNBD server: Celeron 1.3G, 256M, “gate”
  - GFS node: 2 * P4 2.4G (no HT), 768M, “lab1/2”
  - Network: TP-LINK Gigabit-Ethernet Switch, r8169
- Software
  - All on Fedora Core 4 with latest updates
    - Kernel 2.6.12-1.1447_FC4
    - CCS 1.0.0-1, CMAN 1.0.0-1, Fence 1.32.1-1
    - DLM 1.0.0-3, GULM 1.0.0-2
    - GFS 6.1.0-3, GNBD 1.0.0-1

Setting Up

- On each node:
  - modprobe gfs
  - modprobe lock_dlm
  - service ccsd start
  - cman_tool join -w
  - fence_tool join -w
- GNBD server:
  - gnbd_serv
  - gnbd_export -d /dev/hda4 -e test

Setting Up

- GFS nodes:
  - modprobe gnbd
  - gnbd_import -i gate
  - gfs_mkfs -p lock_dlm -t alpha:test -j 2 /dev/gnbd/test
  - mount -t gfs -o noatime /dev/gnbd/test /mnt

Raw Performance

- UDP speed with ttcp
  - All can up to 54.9MB/s
  - sys load near 100% on gate, near 40% on labX
- TCP speed with ttcp
  - 41.0MB/s gate to labX, 47.4MB/s labX to gate, 49.5MB/s between lab1 and lab2
  - sys load 100% on gate, above 60% on labX

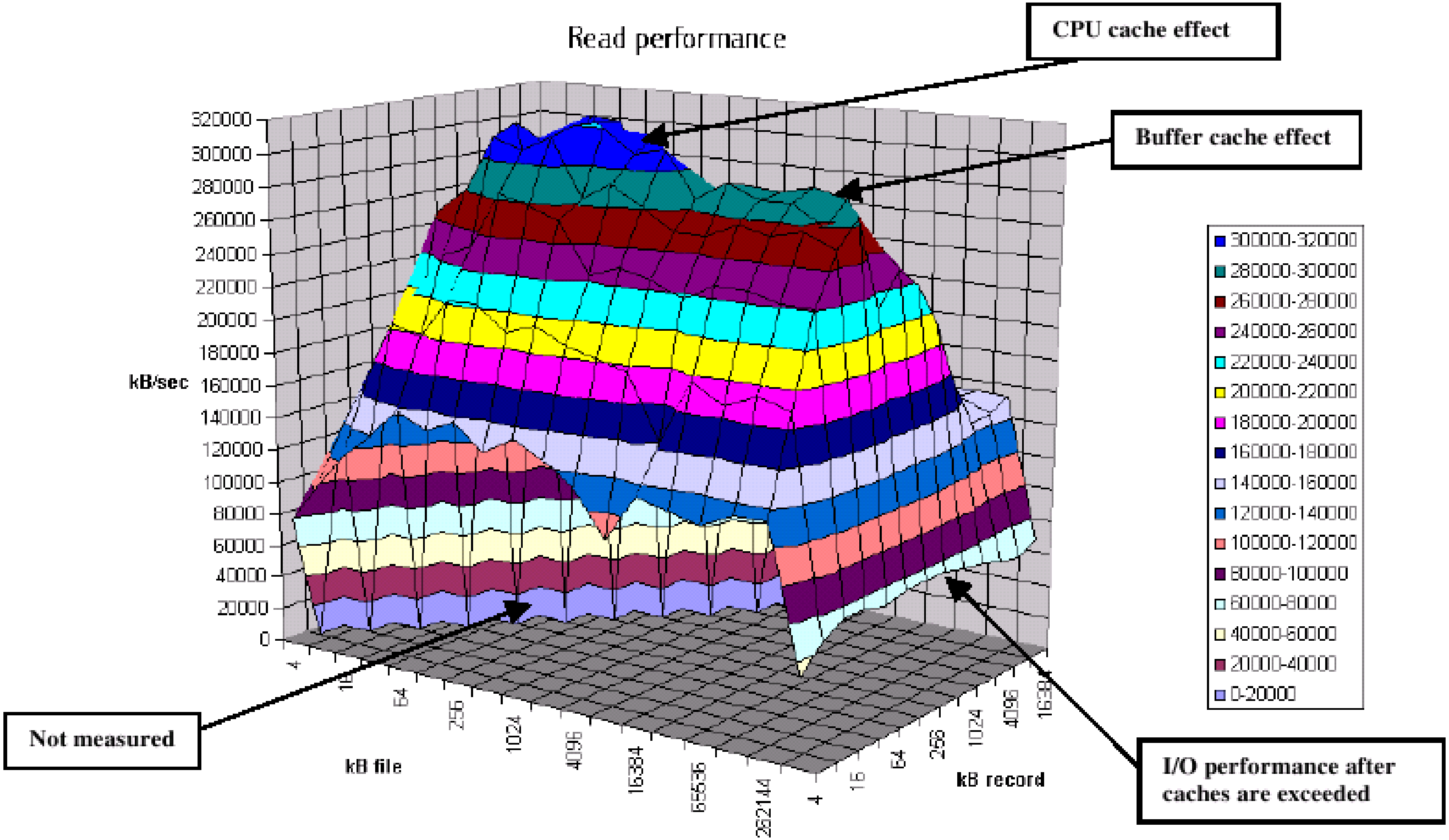
Raw Performance

- DD read GNBD
  - 400MB takes 17.581s, means 22.75MB/s
  - Network traffic 24.6MB/s
- DD read GFS
  - 400MB takes 18.541s, means 21.57MB/s
  - Network traffic 22-24MB/s

Performance Test

- Software
  - iozone
- Issues
  - Locks and local caches
  - GNBD server can be a bottleneck...
  - Need more nodes for scalability
  - Lack of cluster FS benchmark tool (e.g., concurrent access of a same file/directory)

File System Benchmark



CPU cache effect

Buffer cache effect

Not measured

I/O performance after caches are exceeded

Benchmark Result

- 10M file, 4K block size, nproc=4

	Single	Concurrent-1	Concurrent-2	Concurrent-s	Rate
write	37171.8	38619.95	16681.37	55301.32	1.49
re-write	50710.27	38462.35	35151.58	73613.93	1.45
read	355531.23	231989.97	524793.2	756783.17	2.13
re-read	340527.62	302924.08	262806.41	565730.49	1.66
reverse-r	372382.53	332292.26	302122.9	634415.16	1.7
stride-r	308746.94	276470.95	202644.41	479115.36	1.55
random-r	315649.12	188664.97	312509.92	501174.89	1.59
mixed	132150.86	149557.21	121815.98	271373.19	2.05
random-w	42137.24	46925.16	41440.06	88365.22	2.1
pwrite	28819.46	27465.62	15104.1	42569.72	1.48
pread	240757.17	415291.35	258297.22	673588.57	2.8

Benchmark Result

- 10M file, 256K block size, nproc=4

	Single	Concurrent-1	Concurrent-2	Concurrent-s	Rate
write	82278.34	82607.55	94686.68	177294.23	2.15
re-write	75179.66	67534.04	70850.11	138384.15	1.84
read	208125.55	189989.38	149383.22	339372.6	1.63
re-read	173750.42	158556.76	167218.15	325774.91	1.87
reverse-r	191646.82	172081.78	157277.09	329358.87	1.72
stride-r	178582.98	159970.76	154513.99	314484.75	1.76
random-r	189360.96	196201.11	168546.33	364747.44	1.93
mixed	129721.5	105879.21	133651.91	239531.12	1.85
random-w	99842.45	87507.81	112835.73	200343.54	2.01
pwrite	60574.36	66073.25	50079.01	116152.26	1.92
pread	151060.63	215106.36	149002.53	364108.89	2.41

Benchmark Result

- 100M file, 4K block size, nproc=4

	Single	Concurrent-1	Concurrent-2	Concurrent-s	Rate
write	14170.67	41045.99	9937.07	50983.06	3.6
re-write	26800.22	32111.41	34748.53	66859.94	2.49
read	248997.67	268973.57	277366.66	546340.23	2.19
re-read	248201.63	260505.45	292916.52	553421.97	2.23
reverse-r	204804.96	209029.41	232283.94	441313.35	2.15
stride-r	196884.8	196157.48	217332.63	413490.11	2.1
random-r	187543.75	193080.32	209276.93	402357.25	2.15
mixed	95633.69	135064.49	101265.9	236330.39	2.47
random-w	33644.09	45225.22	47585.19	92810.41	2.76
pwrite	19929.62	21812.67	19102.5	40915.17	2.05
pread	248887.67	252778.22	277801.1	530579.32	2.13

Benchmark Result

- 100M file, 256K block size, nproc=4

	Single	Concurrent-1	Concurrent-2	Concurrent-s	Rate
write	60106.44	52114.33	13495.73	65610.06	1.09
re-write	52896.39	54377.92	49747.09	104125.01	1.97
read	167019.93	177912.24	143083.77	320996.01	1.92
re-read	167236.99	172567.24	137500.43	310067.67	1.85
reverse-r	160100	173721.75	138457.31	312179.06	1.95
stride-r	166677.08	174354.42	136507.06	310861.48	1.87
random-r	167970.22	172542.45	135181.97	307724.42	1.83
mixed	97057.91	107903.34	126889.17	234792.51	2.42
random-w	67654.24	61172.31	101672.84	162845.15	2.41
pwrite	42152.71	37985.71	55781.75	93767.46	2.22
pread	170737.26	171491.16	131561.04	303052.2	1.77