Systems & Technology Group

GPFS: General Parallel File System

Gautam Shah

gautam@us.ibm.com

Scalable I/O Development

IBM Poughkeepsie Development Lab

IBM Confidential

© 2006
IBM Corporation

Notice

- **The traditional Disclaimer:**
  - The plans presented are subject to change without prior notice.
  - The material is IBM Confidential and covered under appropriate Non-Disclosure Agreements.

Outline

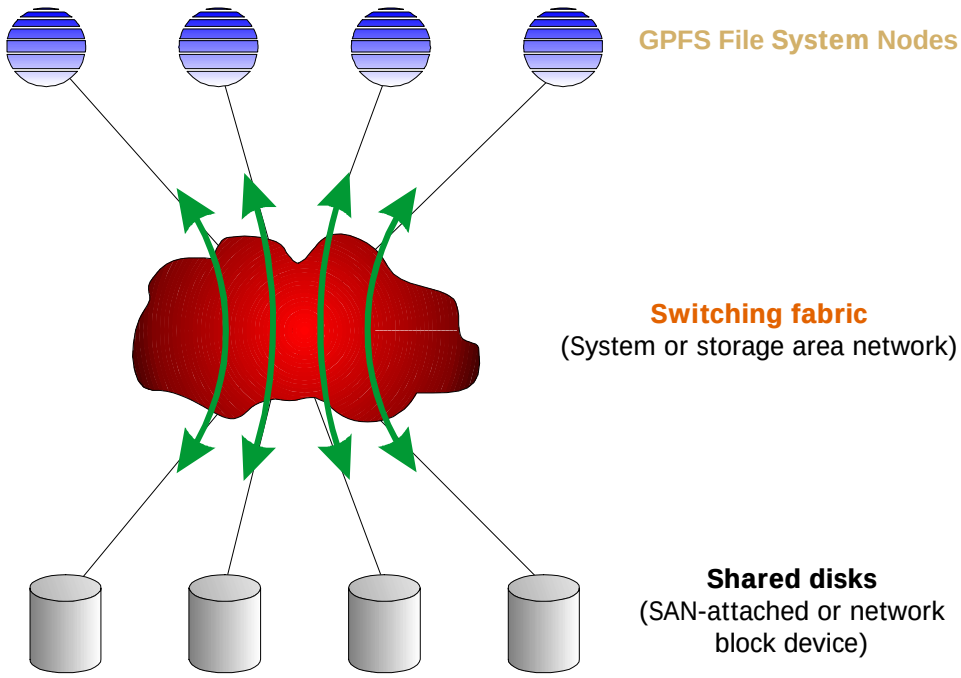
- **What is GPFS?**
- **Why do we need GPFS?**
- **GPFS Features**
  - Basic Architecture
  - Interoperability
  - Multi-cluster
  - ILM features
  - Other features
- **Future Directions**

IBM Confidential; subject to change without notice

What is GPFS?

Parallel File System for Cluster Computers Based on Shared Disk (SAN) Model

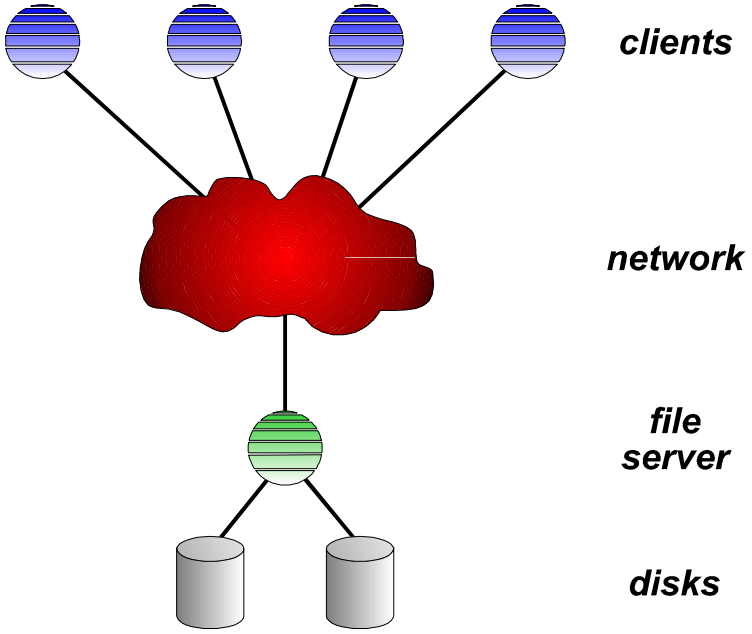
- ***Cluster –*** **a collection of fabric-interconnected nodes (IP, SAN, …)**
- ***Shared disk -*** **all data and metadata on fabric-attached disk**
- ***Parallel -*** **data and metadata flows from all of the nodes to all of the disks in parallel under control of distributed lock manager.**

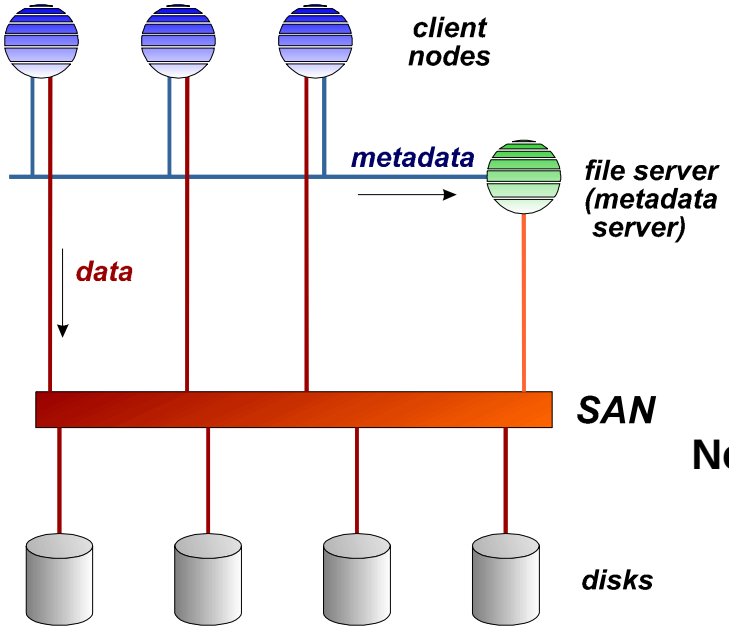


IBM Confidential; subject to change without notice

What GPFS is *NOT*

Not a client-server file system like NFS, DFS, or AFS: no single-server bottleneck, no protocol overhead for data transfer



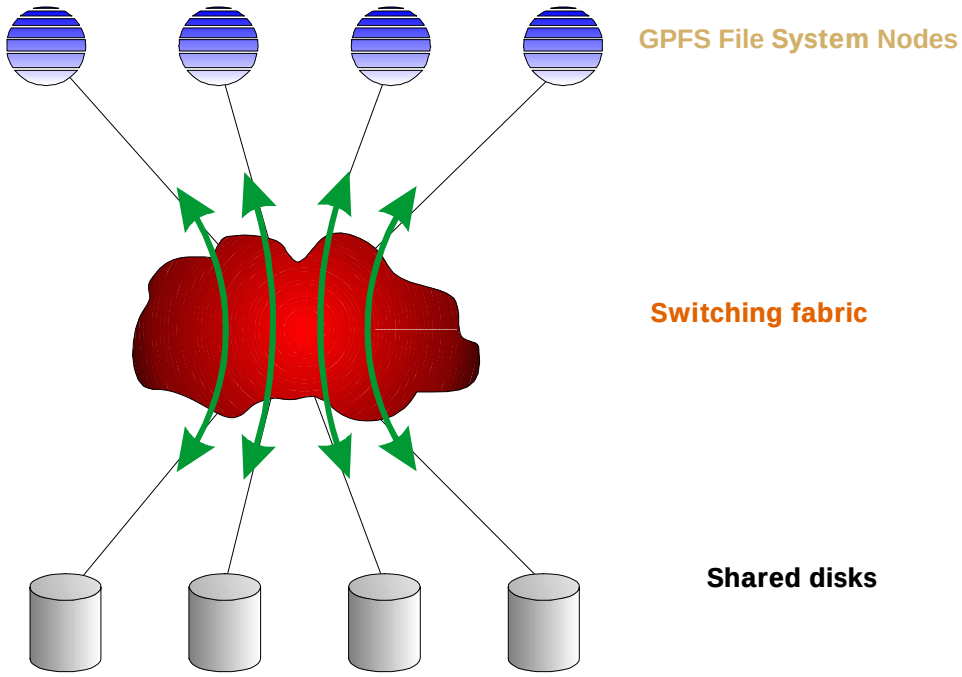


Not like some SAN file systems (e.g. Veritas, CXFS): no distinct metadata server (which is a potential bottleneck)

IBM Confidential; subject to change without notice

GPFS: General Parallel File System

- **Available since 1998 (AIX), on Linux since 2001.**
- **Product available on pSeries and xSeries (IA32, IA64, Opteron), on AIX and Linux, and on Blue Gene.**
- **Also runs on compatible non-IBM servers and storage.**
- **Thousands of installs, including many Top 500 supercomputers**
- **Customers use GPFS in many applications**
  - High-performance computing
  - Scalable file and Web servers
  - Database and digital libraries
  - Digital media
  - Analytics, financial data management, engineering design, …



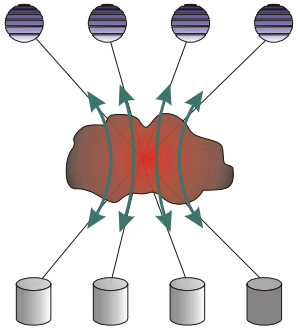
IBM Confidential; subject to change without notice

Distinguishing Features

- **Performance for data intensive applications. 120+ GB/sec in recent ASC Purple delivery.**
- **File serving scalability via NFS and Samba.**
- **Scalability. 2400 nodes deployed at Mare Nostrum**
- **Fault tolerance.**
- **Disaster recovery facilities through replication**
- **Interfaces for HSM exploited by TSM and HPSS**
- **SAN and general network access to disks**
- **Multi-clustering capabilities**
- **Blue Gene**
- **AIX/Linux Support**

IBM Confidential; subject to change without notice

Why is GPFS needed?



Clustered applications impose new requirements on the file system

- **Applications need high-performance access to their data from wherever they run**
  - Parallel applications also need fine-grained access within a file from multiple nodes
  - Serial applications dynamically assigned to processors based on load
- **Applications require good availability of data and normal file system semantics**

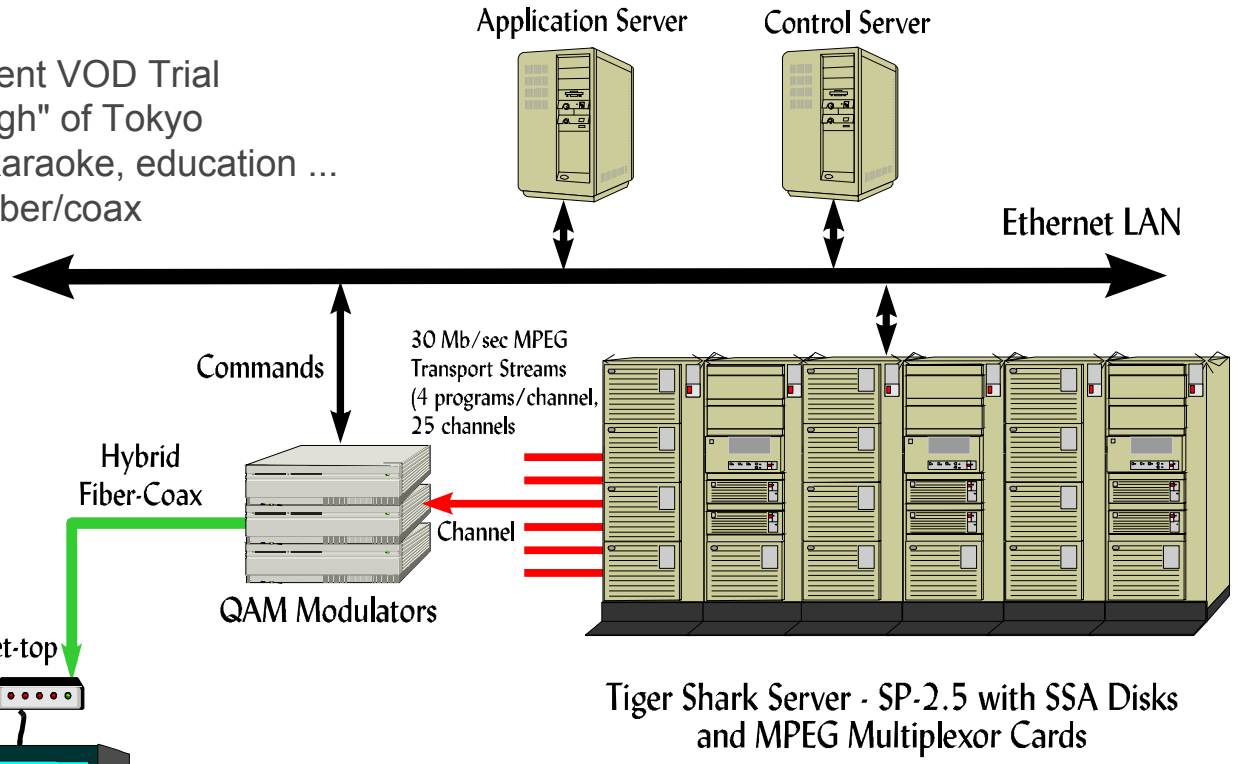
GPFS supports this via:

- **uniform access – single-system image across cluster**
- **conventional POSIX interface – no program modification**
- **high capacity – multi-TB files, petabyte file systems**
- **high throughput – wide striping, large blocks, many GB/sec to one file**
- **parallel data and metadata access – shared disk and distributed locking**
- **reliability and fault-tolerance - node and disk failures**
- **online system management – dynamic configuration and monitoring**

IBM Confidential; subject to change without notice

Digital Media

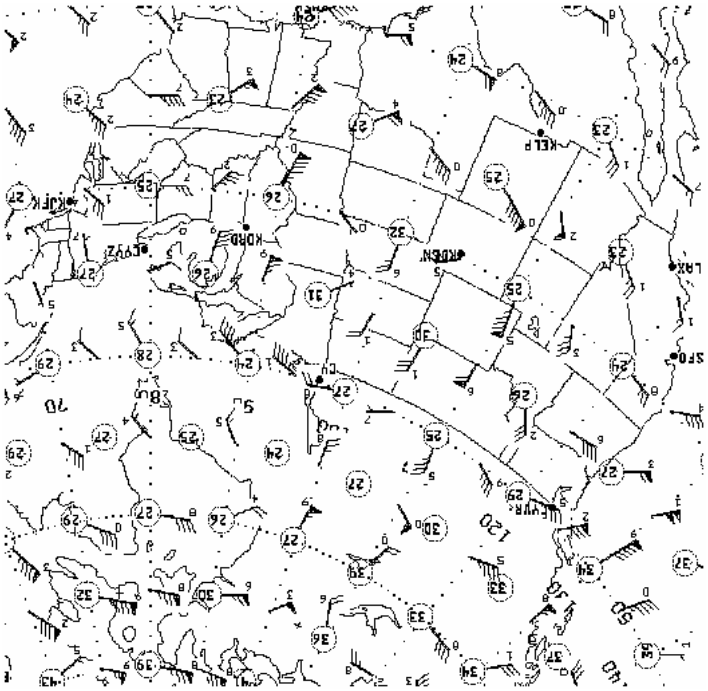
- Tokyo Metropolitan Government VOD Trial
- Interactive TV for new "borough" of Tokyo
- Applications: movies, news, karaoke, education ...
- Video distribution via hybrid fiber/coax
- Trial went "live" June '96
- 500 subscribers



- 6 Mbit/sec MPEG video streams
- 100 simultaneous viewers (75 MB/sec)
- 200 hours of video on line (700 GB)
- 12-node SP-2 (7 distribution, 5 storage)

IBM Confidential; subject to
change without notice

Government Laboratories

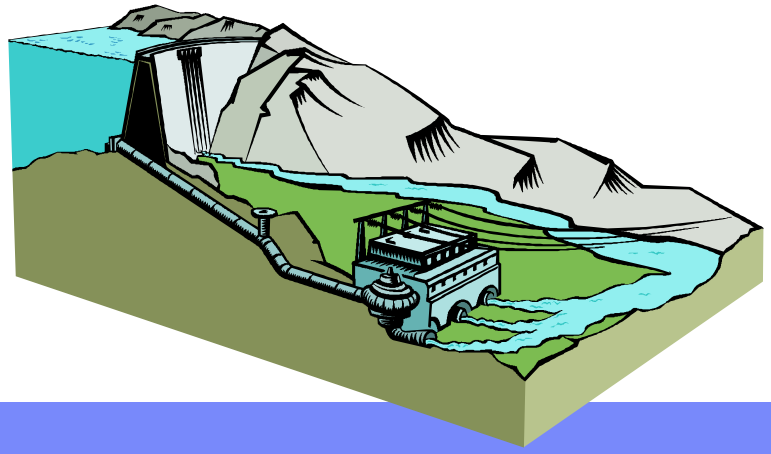


NOAA - NCEP, NCAR

- Application: large-scale, highly precise weather modeling
- 400-node SPs
- Using GPFS to store captured observation data used as input to weather models and to store output for subsequent visualization

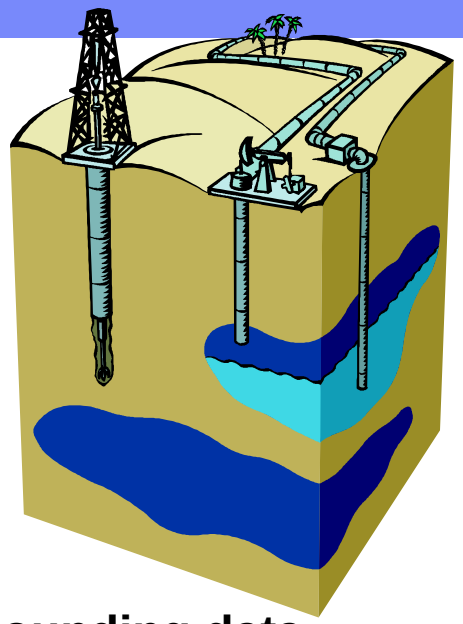
Army Corps of Engineers

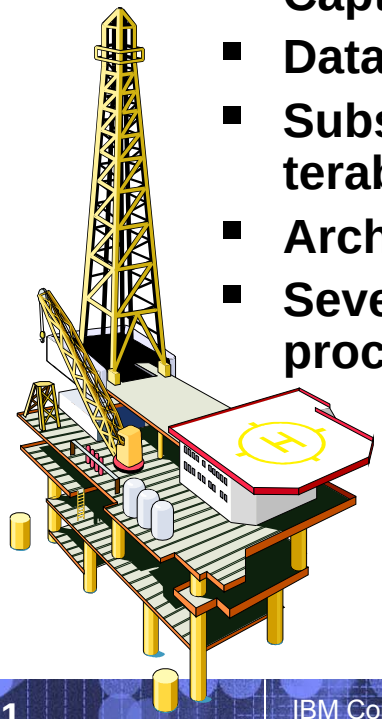
- **Application: hydrological modeling**
- **2 256-node SPs**
- **Using GPFS to store model data, output visualization data**



IBM Confidential; subject to change without notice

Geophysical





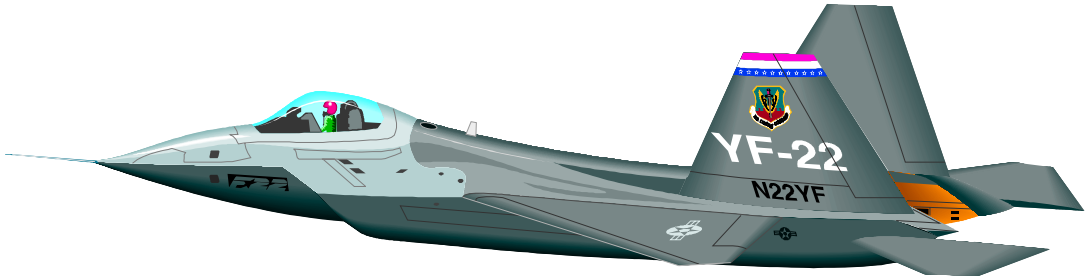
- **Capture, storage, and analysis of seismic sounding data**
- **Data captured shipboard on small SP (sequential access)**
- **Subsequent analysis at home base (highly parallel, multi-terabyte file systems)**
- **Archive/retrieval from tape farm**
- **Several geophysical firms using GPFS for seismic processing**

IBM Confidential; subject to change without notice

Engineering Design

Aircraft manufacturer

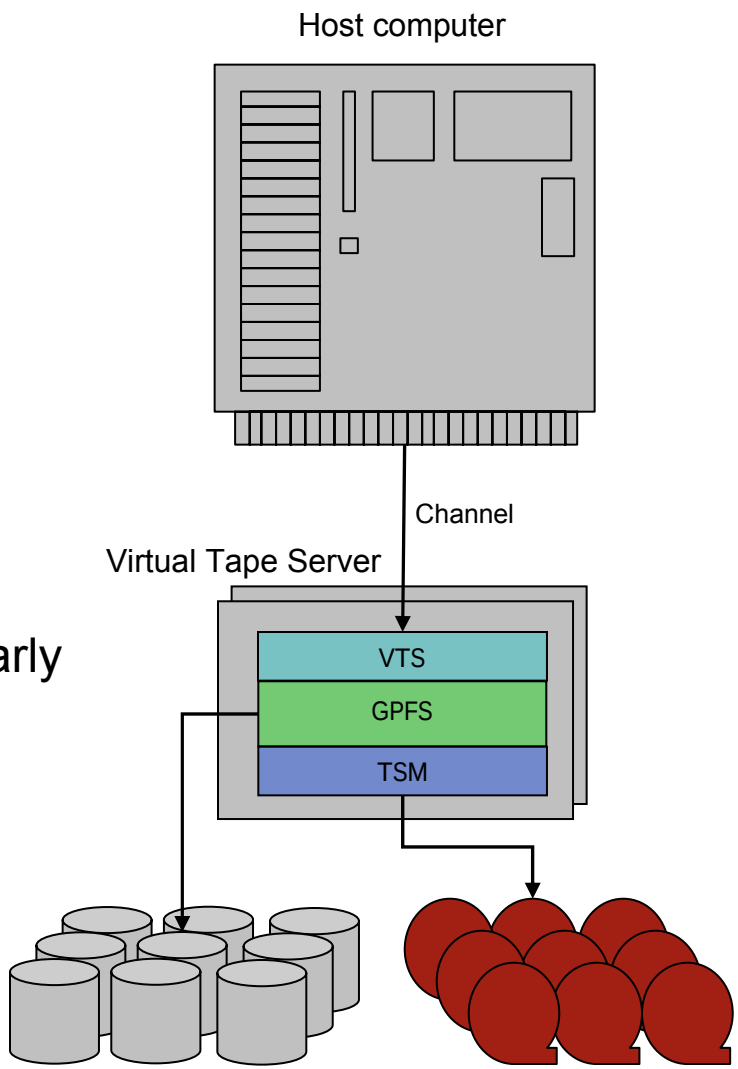
- **Using GPFS to store large designs for structural modeling and analysis**
- **GPFS allows all nodes to share designs and models**



IBM Confidential; subject to change without notice

IBM Virtual Tape Server

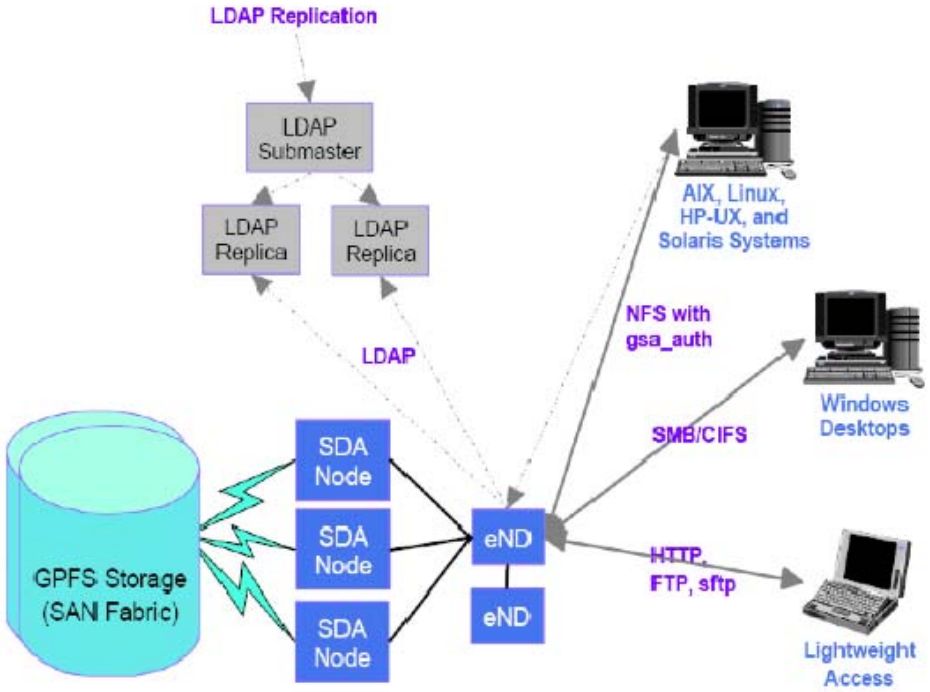
- GPFS is a major component of the IBM Virtual Tape Server (VTS)
  - VTS control component emulates tape to host computer
  - GPFS provides high-performance speed-matching buffer and cache on disk
  - TSM provides tape management
- GPFS provides 5x speedup over local AIX file system when using RAID disks, improves total system performance 2x
- GPFS initially offered as VTS Turbo feature in early VTS release
- Later releases use GPFS as a standard part of VTS
- Peer-to-Peer VTS features multiple controllers and libraries for increased capacity and availability



IBM Confidential; subject to change without notice

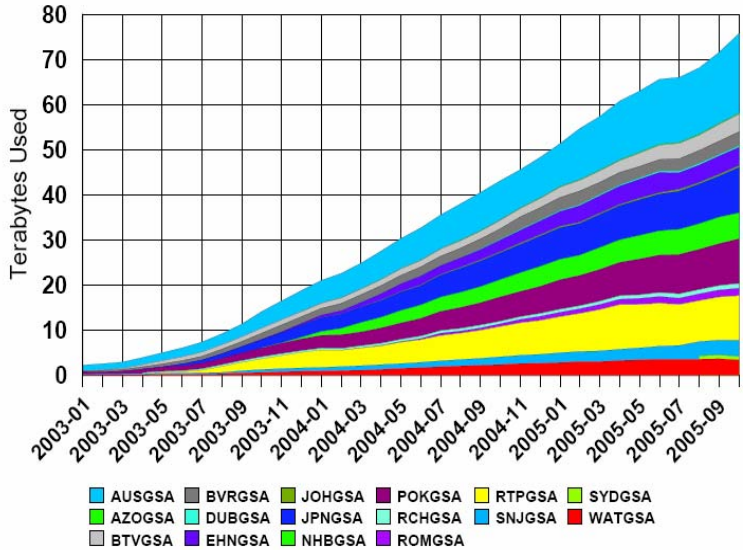
GSA Experience

GSA File cell diagram



GSA File world wide availability - last 12 months

GSA File space usage trends by cell

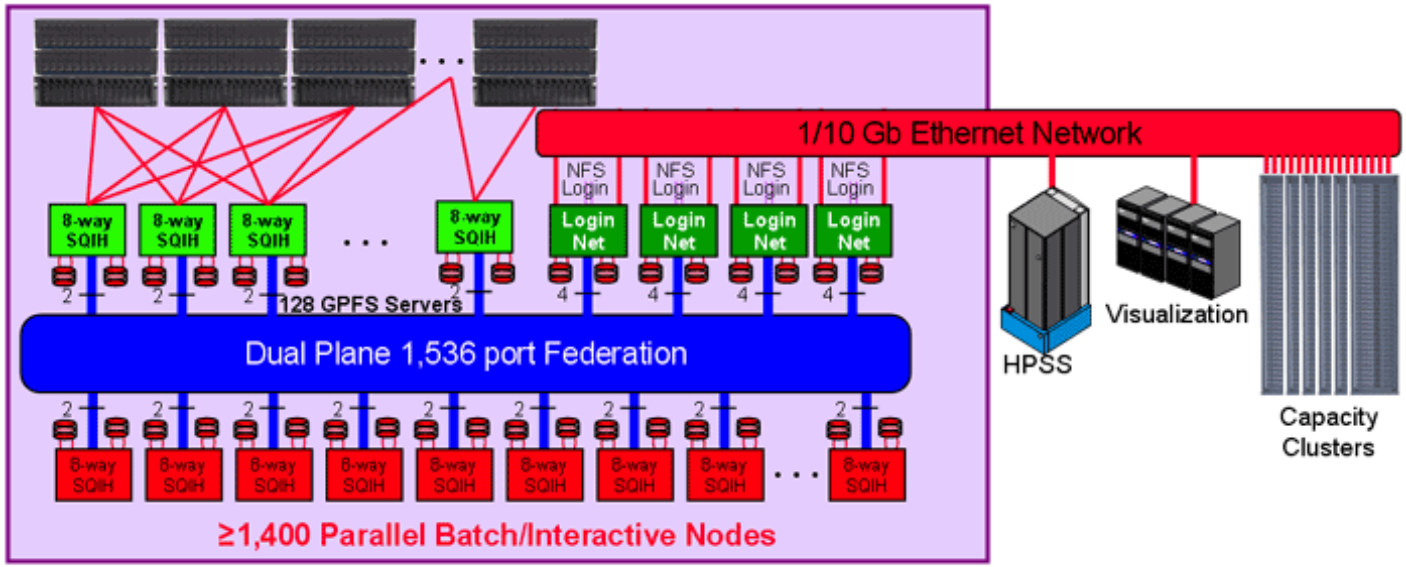


August, 2006

142TB total space

IBM Confidential; subject to change without notice

GPFS on ASC Purple/C Supercomputer



Purple System
- At least 1,400 parallel batch/interactive nodes
- 4 Login/network nodes from 2 SQH
- Clustered I/O with 128 SQIH for global I/O
- Dual plane 1,536 port Federation switch
- External networking
  - Login/network nodes for login/NFS/PFTP
  - All external networking is 1-10Gb/s Ethernet

Programming/Usage Model
- Application launch over all compute nodes
- 1 MPI task/CPU and Shared Memory, full 64b support
- Scalable MPI (MPI_allreduce, buffer space) to 8,192 tasks
- Likely usage
  - multiple MPI tasks/node with 2-4 OpenMP/MPI task
- Single STDIO interface
- Parallel I/O to single file, multiple serial I/O (1 file/MPI task)

- **1536-node (128 I/O nodes), 100 TF pSeries cluster at Lawrence Livermore National Laboratory**
- **2 PB GPFS file system (one mount point)**
- **500 RAID conroller pairs, 11000 7200RPM SATA disk drives**
- **130 GB/s parallel I/O measured to a single file (135GB/s to multiple files)**

IBM Confidential; subject to change without notice

Performance

- **130+ GB/s parallel read/write on achieved on ASCII Purple**
- **12 GB/s parallel read/8 GB/s write on 40-node IA64 Linux cluster with 120 TB SAN-attached FAStT 600**
- **7.8GB/s streaming read, 7.3GB/s write to single file on p690**
  - 15.8 GB/s read/14.5 GB/s write aggregate to multiple files
- **1000+ files/sec/node (3051 files/sec on 3 Linux nodes, 50 drives on one FAStT controller (256M files in 24 hours 26 mins).**

IBM Confidential; subject to change without notice

GPFS Architecture

High capacity:

- Large number of disks in a single FS

High BW access to single file

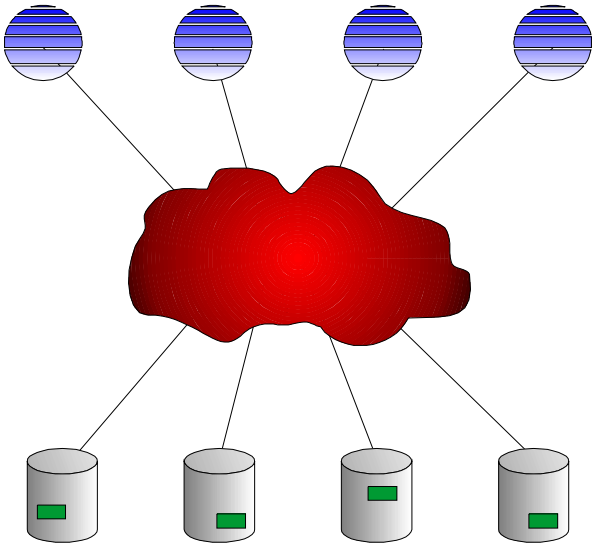
- Large block size, full-stride I/O to RAID
- Wide striping – one file over all disks
- Multiple nodes read/write in parallel

High availability

- Nodes: log recovery restores consistency after a node failure
- Data: RAID or internal replication
- On-line management (add/remove disks or nodes without un-mounting)

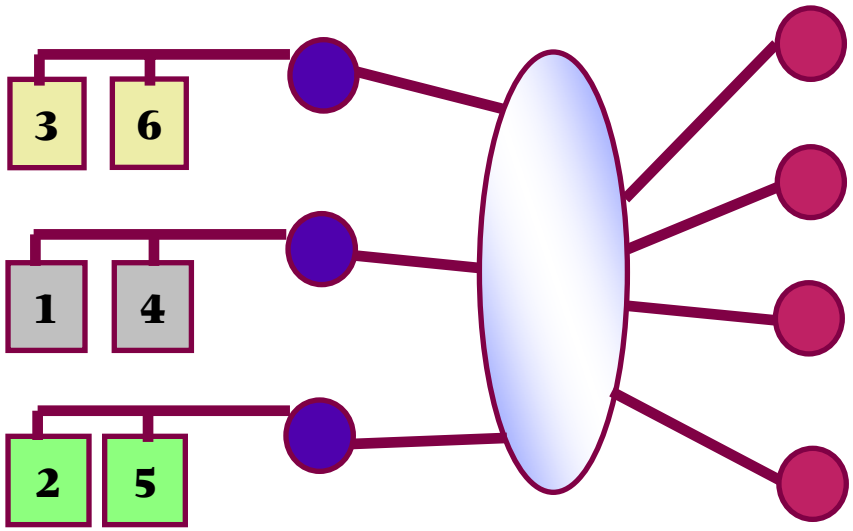
Single-system image, standard POSIX interface

- Distributed locking for read/write semantics



IBM Confidential; subject to change without notice

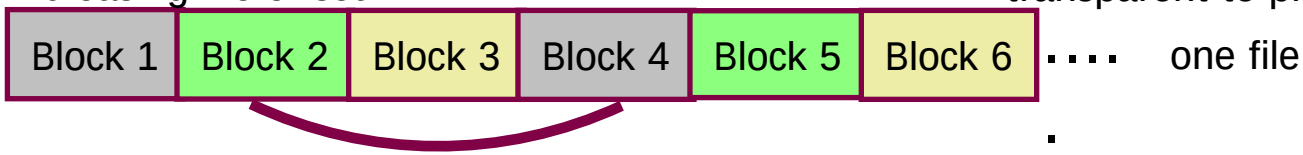
Data Striping



Disk Pool | Server Nodes | SP Switch | Application Nodes

- GPFS *stripes* successive blocks of each file across successive disks
- Disk I/O for sequential reads and writes is done in parallel (prefetch, write behind)
- Make no assumptions about the striping pattern
- Block size is configured when file system is configured, and is *not* programmable
- choices are 16, 64, 128, 256, 512, 1024 KB , 2 MB, 4 MB (most common choice = 256 KB)
- transparent to programmer

increasing file offset ---->



.... one file

3 I/Os executed in parallel

Job reads at 30 MB/s

Each disk reads at 10 MB/s

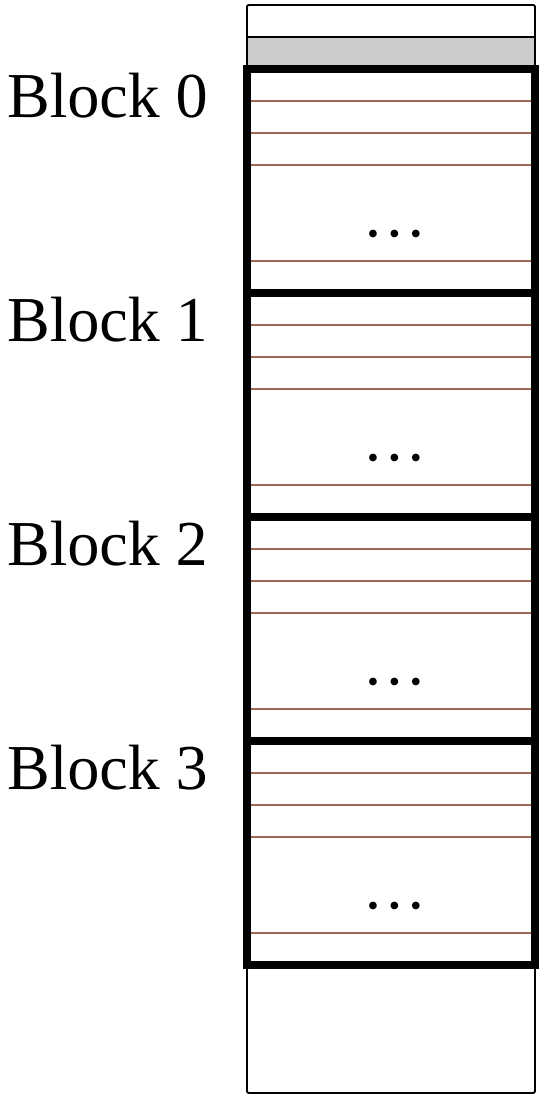
IBM Confidential; subject to change without notice

GPFS: Blocks and Sub-blocks

- **A GPFS file system can be configured with one of the following block sizes**
  - 4 MB
  - 2 MB
  - 1 MB
  - 512 KB
  - 256 KB
  - 128 KB
  - 64 KB
  - 16 KB
- **Large block sizes optimize performance when large record accesses are common (by reducing the number of IOPs)**
- **Small block sizes optimize storage when small record accesses are common**
- **In this presentation, assume block size is 256 KB unless otherwise stated**
- **Blocks can be divided into 32 sub-blocks**
  - A block is the largest chunk of contiguous data that can be accessed
  - A sub-block is the smallest chunk of contiguous data that can be accessed
  - Files smaller than a block can be stored in fragments of 1 or more sub-blocks (common in life science applications)
  - If sizeof(record) <= sizeof(subblock), then only a sub-block is accessed, but if 2 or more sub-blocks are accessed in block, then the entire block is accessed

IBM Confidential; subject to change without notice

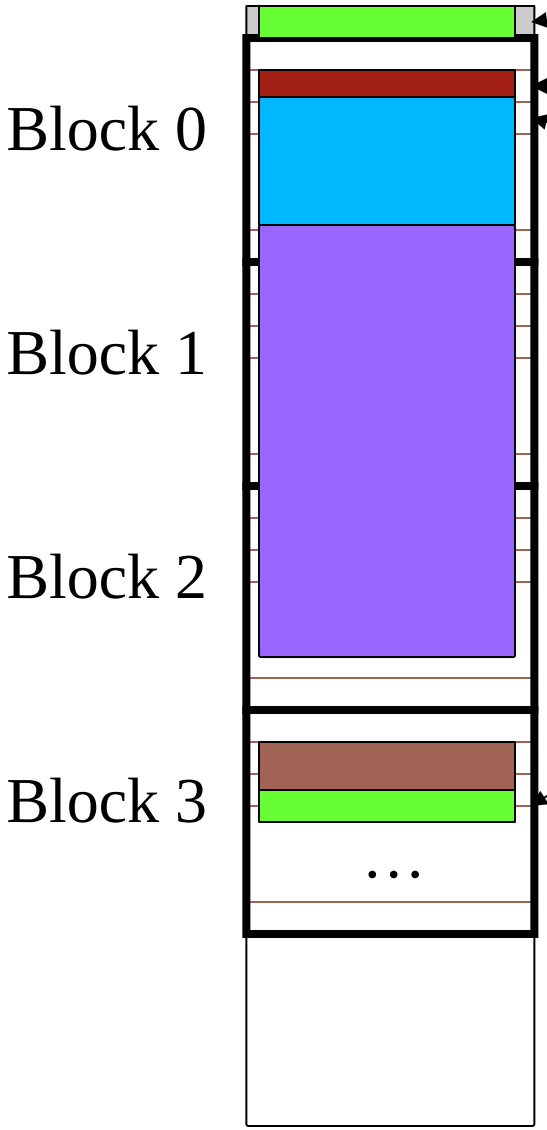
Basic Disk Layout



- **Disk divided into fixed size "blocks" (supported sizes: 16k, 64k, 128k, 256k, 512k, 1M, 4M)**
- **Each block divided into 32 "subblocks" (512 – 32k)**
- ***Block*: largest contiguously allocated unit (largest unit read/written in single I/O)**
- ***Subblock*: smallest allocatable unit**

IBM Confidential; subject to change without notice

Disk Layout: Fixed Metadata



Sector 0: not used by GPFS

Sector 1: disk descriptor

Sector 2: gpfs disk identifier ("NSD-ID")

Sector 3-7: reserved

Sector 8 – 4089: File System Descriptor (super block)

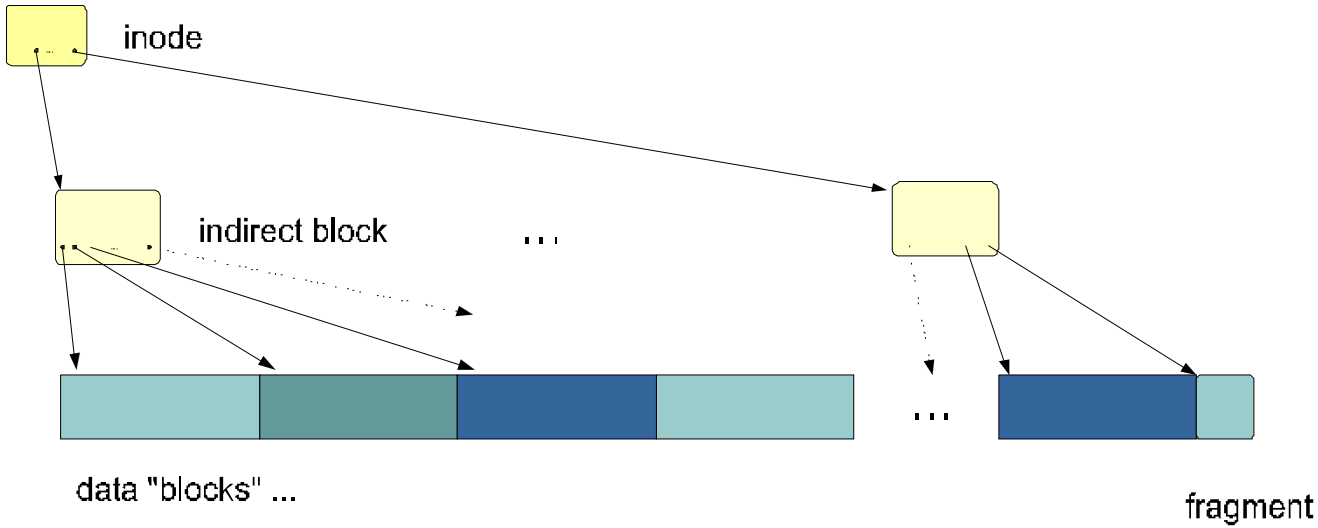
Sector 4090 – 4097: Paxos sectors

Sector 4098: inode 0

All other data and metadata stored as "files" in blocks spread over all available disk in a file system

IBM Confidential; subject to change without notice

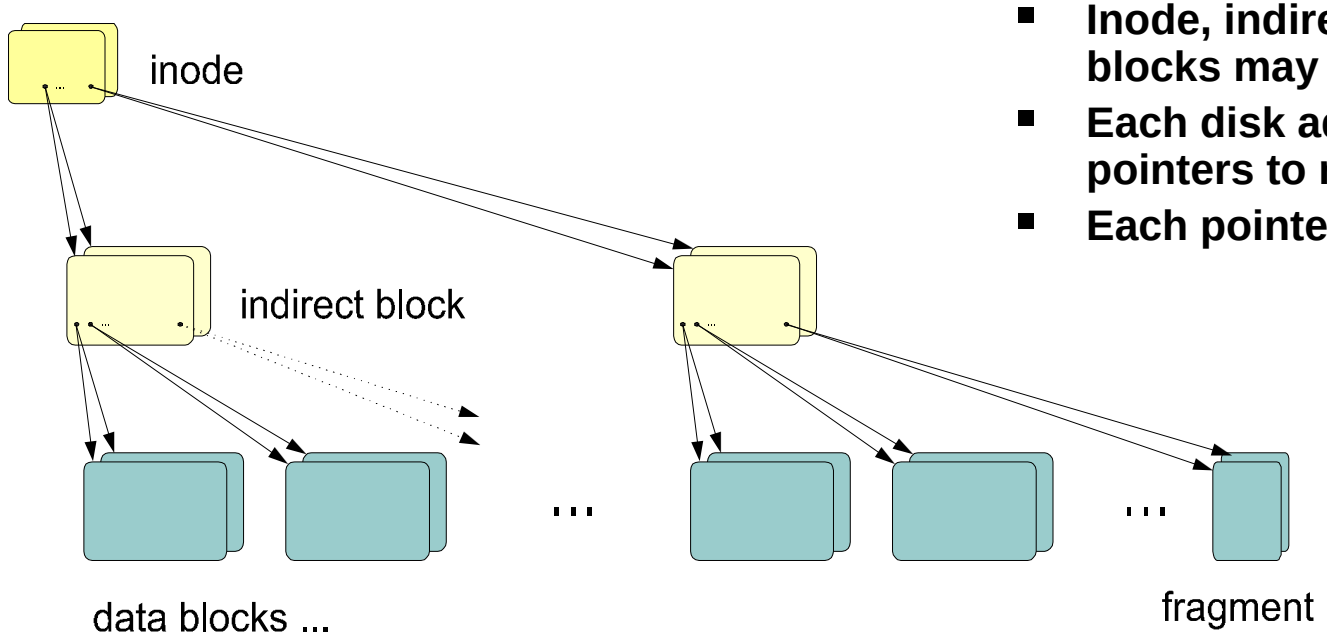
Disk Structures: File Data



- **File data divided into fixed-size extents or “data blocks” (default 256k)**
- **Data blocks striped round-robin across all disks**
- **Last block may be a fragment;**
  reduces space overhead, especially for small files
- **Individual block stored at random location within assigned disk**
- **Inode & indirect blocks like traditional Unix file systems**

IBM Confidential; subject to
change without notice

GPFS Replication



- **Inode, indirect block, and/or data blocks may be replicated**
- **Each disk address: list of pointers to replicas**
- **Each pointer: disk id + sector no.**

- **No designated "mirror", no fixed placement function:**
- **Replicas stored in separate *failure groups***
  - *Failure group* is a set of disks with a common failure mode
  - … e.g. a brick.
- **Replicas spread evenly over disks for load balancing**
- **Dynamic reconfiguration: data can migrate block-by-block**

IBM Confidential; subject to change without notice

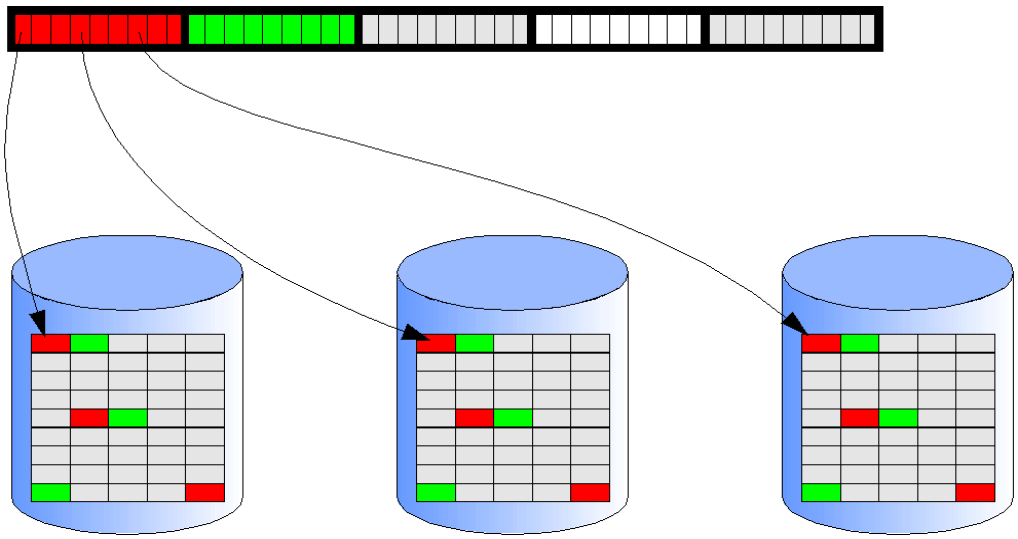
Special Files

- **Metadata files (aka. “low-level files”):**
  - Inode file
  - Block allocation map
  - Inode allocation map
  - ACL file
  - Extended attribute file
  - Log files
  - Allocation summary file
- **Same on-disk structure as regular files**

IBM Confidential; subject to change without notice

Allocation maps

- Segmented Block Allocation MAP:



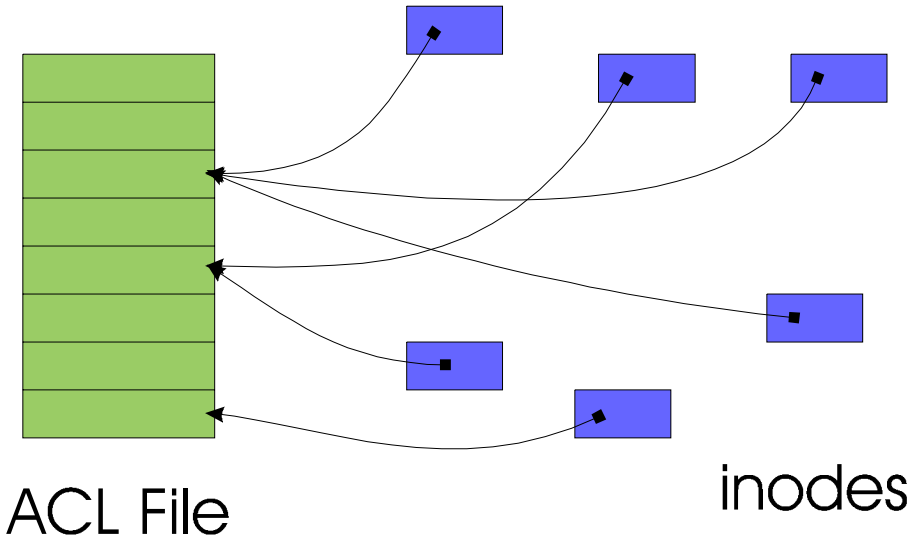
- Each segment contains bits representing blocks on all disks
- Each segment is a separately lockable unit
- Minimizes conflicts between multiple writers
- Allocation manager provides hints which segments to try

Similar: inode allocation map

IBM Confidential; subject to change without notice

Access Control Lists (ACLs)

- **POSIX ACLs, NFSV4 ACLs**
- **All ACLs stored in a single file**
- **Multiple inodes can refer to the same ACL**
- **ACL sharing: efficient retrieval and caching; efficient inheritance**
- **Unused ACL space reclaimed via garbage collection**



- **ACL updates: High degree of read sharing; updates infrequent, but read/write conflicts expensive. *Solution*: allow multiple readers plus single writer; broadcast ACL updates.**

IBM Confidential; subject to change without notice

Extended Attributes

- **Stored in a sparse file**
- **The *i*'th block of the xattr file contains attributes for file with inode number *I***
- **Current use of extended attributes: DMAPI, direct-IO**

IBM Confidential; subject to change without notice

Log Files

- One (replicated) log file per node
- Created as needed at mount time
- Fixed size

IBM Confidential; subject to change without notice

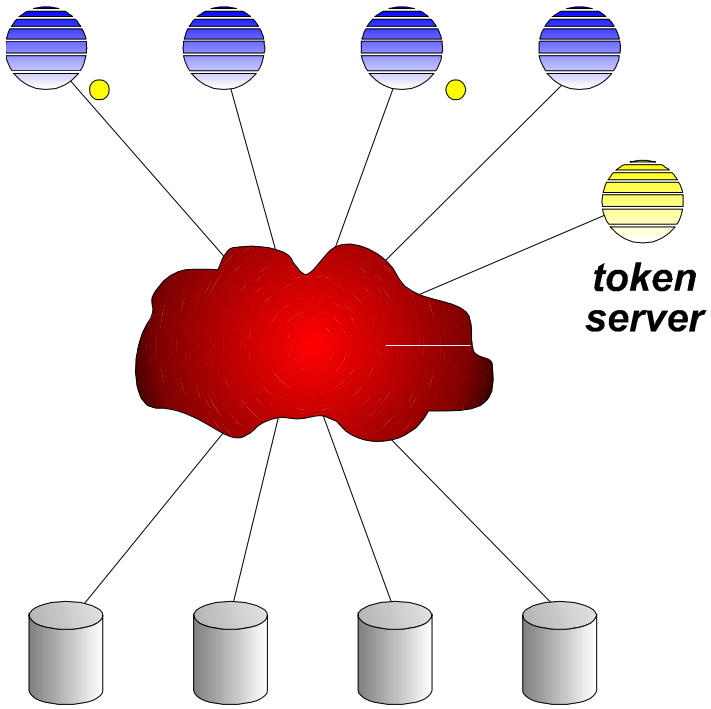
Distributed Locking

- **Distributed locking necessary to …**
  1) *Synchronize* updates to file system *metadata on disk* to prevent corruption
  2) Maintain *cache consistency* of data and metadata cached in memory on different nodes.
  3) Synchronize file system operations for *POSIX semantics*
- **Token-based distributed lock manager:**
  - Allows “lock caching”.
  - Avoid sending a lock message on every single operation

IBM Confidential; subject to change without notice

Token-based Lock Manager

- **Token server grants tokens.**
- **Token represents right to read, cache, and/or update a particular piece of data or metadata.**
- **Single message to token server allows repeated access to the same object.**
- **Conflicting operation on another node will revoke the token.**
- **Force-on-steal: dirty data & metadata flushed to disk when token is stolen.**
- **Numerous optimizations:**
  - Large byte ranges
  - Lookup/open



IBM Confidential; subject to change without notice

Challenge: Achieving high bandwidth

- **Problem: Achieving high bandwidth to a single file requires writing from multiple nodes in parallel.**
- **Solution: Lock byte ranges instead of whole files or individual data blocks [Byte Range Locking]**

  Each token request contains *required range* (current operation) and *desired range* (likely future operations).

  Example: sequential writer:
  desired range = [current write offset, infinity]

  - First writer receives BR token [0, infinity].
  - Second writer: revoke cuts token into two pieces (unless first writer has already closed the file).
  - Each additional writer: revoke cuts one of the remaining pieces in two.
- **Concurrent write sharing: single revoke per node.**
- **Sequential sharing: as efficient as whole file locking.**

IBM Confidential; subject to change without notice

Centralized Metadata Servers

- Parallel user data semantics, but non-parallel metadata semantics
- Metadata maintained and accessed from a single common server
  - Potential bottleneck that can limit scaling
  - Failover features allow a backup metadata server to takeover if the primary fails
- Metadata typically exchanged over Ethernet
  - Slower interconnect technology reduces performance (100 MbE vs GbE vs Myrinet)
  - Cost reduction feature? Other options are possible (e.g., switch)
- All "disks" connected to all nodes
  - do not need expensive switch (e.g., Myrinet, Colony)
  - inhibits scaling due to cost of FC Switch Tree (i.e., SAN)
- Ideal for smaller numbers of nodes
- Example: CXFS (SGI), SANergy (IBM), StorNextFS (ADIC)
- COMMENT: These may be called "parallel like" file systems.

  These products are relatively new; as they mature, some of these limitations may be improved.

IBM Confidential; subject to change without notice

Metadata Updates (MNode)

- **Problem: Under concurrent write sharing, updates to file metadata could become a bottleneck (e.g., file size and mtime in the inode).**
- **Solution:**
  - *Shared write lock* allows concurrent updates to file size and mtime.
  - *Metanode* collects and merges file size and mtime updates (keep largest file size and latest mtime).
- **Operations that require exact file size/mtime (e.g., stat) conflict with the shared write locks.**
- **Operations that may decrease file size or mtime (trunc, utimes) require an exclusive lock.**
- **Metanode is elected dynamically and can move dynamically.**

IBM Confidential; subject to change without notice

The GPFS Metanode

- **One per open file, elected dynamically**
  - Generally, the node which has had file opened longest continuous period of time
- **Roles**
  - Maintain file metadata integrity
  - Performs all metadata I/O for a file (inode, indirect blocks)
  - Collects/merges metadata updates from other nodes

IBM Confidential; subject to change without notice

Allocation Map Updates

- **Problem: Single block allocation map shared among nodes could become a bottleneck.**
- **Solution:**

  Allocation bits for different disks interleaved in the allocation map.
  - Accessing a single block of the allocation map allows allocating space on all disks.

  Allocation manager maintains free space hints.
  - Avoid each node searching the whole map for free space.
  - Prevent lock conflicts by directing different nodes at different allocation map blocks.

IBM Confidential; subject to change without notice

Token Server Scaling

- **Problem: Token server bottleneck, especially with large numbers of small files?**
- **Solution:**
  - Token batching and token pre-fetch reduce token traffic.
  - Token re-use through inode re-use.
  - Avoid lock conflicts through
    - BR locking, metanode, allocation manager
    - Data-shipping, MPI/IO, Direct I/O

    (I/O traffic due to conflicts more significant than token traffic.)
- **Single token server node sufficient even in very large systems.**
- **Distributed Token Manager, new in Version 3.1**

IBM Confidential; subject to change without notice

Configuration Options – Disk Leasing

- **Disk Leasing**
  - Heartbeating
  - A mechanism to prevent a node from writing to a disk when it should not, e.g., perceived node failure
  - If there exist valid globally attached NSD's in a cluster, this cluster will automatically operate in the disk leasing mode.
  - In this mode, an active GPFS daemon is allowed to perform I/O operation only if it holds a valid lease.
  - Disk lease duration and renewals are managed by the GPFS configuration manager node.

IBM Confidential; subject to change without notice

Configuration Options – Quorum

- Quorum is used to prevent a cluster getting partitioned. If this were to happen then each partition would have a unique file system manager and the state of the file system would become inconsistent.
- Normal Quorum Rule:
  - number_quorum_nodes/2 + 1 = quorum minimum

IBM Confidential; subject to change without notice

Configuration Options – Networks

- **Node Network Connectivity**
  - Network Types
    - Myrinet
    - Gb Ethernet
    - 10 Gb Ethernet
    - IB
    - IBM HPS
  - Large clusters can have separate subnets for login and data/private networks (can be of different speeds and types)
  - GPFS traffic comprises two types of traffic
    - Token/Admin traffic – GPFS daemon/administrative data
    - Data traffic – User data
  - tscSubnet [new in version 3.1]

IBM Confidential; subject to change without notice

Configuration Options – Ethernet/Channel Bonding

- **Node Network Connectivity**
  - Channel bonding (or link aggregation) of ethernet adapters can deliver increased levels of performance and a higher level of availability.
  - Nodes with different roles will have different networking requirements. NSD servers may need multiple interfaces to deliver bandwidth while clients may only require a single adapter.

IBM Confidential; subject to change without notice

GPFS Cluster Size Limits

GPFS for Linux on Multiplatform	**2441 nodes[1]**
GPFS for AIX 5L on POWER	**1530 nodes[2]**

Contact gpfs@us.ibm.com

1. Configurations larger than 512 nodes require a review by IBM
2. Configurations larger than 128 nodes require a review from IBM

IBM Confidential; subject to change without notice

GPFS File System Size Limits

GPFS 2.3, or later, Architectural File System Size Limit	**2 ^ 99 bytes**
Current Tested Limit	**Approximately 2 PB**

Contact gpfs@us.ibm.com if you intend to exceed 200 Terabytes

Architectural limit on number of files in a file system is 2,147,483,648. Effective limit usually lower than architectural limit, and could be adjusted using “-F” option on the mmchfs command.

IBM Confidential; subject to change without notice

GPFS Cluster Size Limits

Total number of nodes that may concurrently join a cluster **A Node is in a cluster IF:** ▪**The node shows in mmlscluster [shows up in mmlscluster] ,or** ▪**The node is in a remote cluster and is mounting a filesystem in the local cluster.**	**4096 Nodes**

IBM Confidential; subject to change without notice

Maximum Number Mounted Filesystems in a GPFS Cluster

GPFS Version 3.1.05, or later	**64**
GPFS Version 3.2	**256**

IBM Confidential; subject to change without notice

GPFS Disk Size Limits

GPFS 3.1 Maximum Disk Size Limit	**Limited by disk device driver and O/S**
AIX 5L with 64 bit kernel	**> 2 TB[1] (up to O/S limit)**
Other supported platforms	**<= 2 TB**

1. Provided the disk device supports it

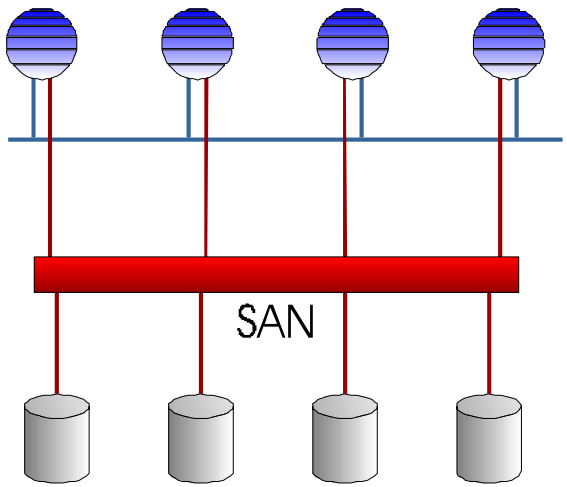
IBM Confidential; subject to change without notice

Storage connectivity

- **SAN's do not scale (at least not today)**
  - Too many hosts cause device queue overflow
  - Even director-class switches have bottlenecks
- **Need other connectivity technology**
  - Limit SAN to back-end switch
  - Route storage traffic onto IP or other fabric (e.g. Infiniband, Federation, …
  - RDMA to offload CPU
  - i.e. "software shared disk"

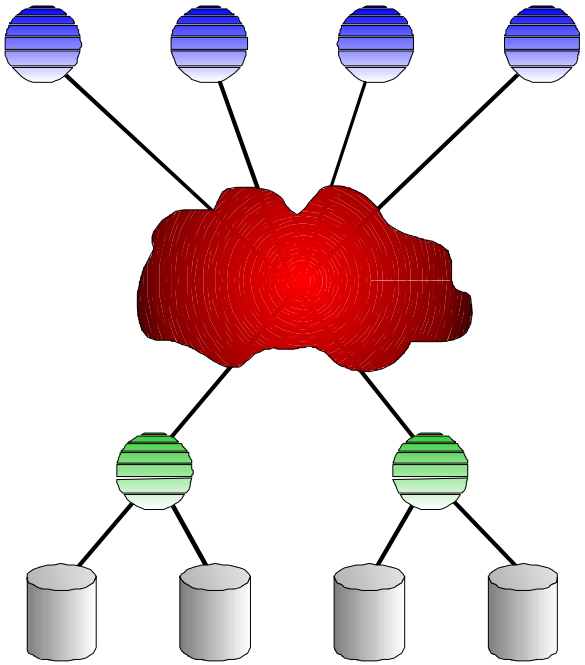
IBM Confidential; subject to change without notice

Configuration Examples



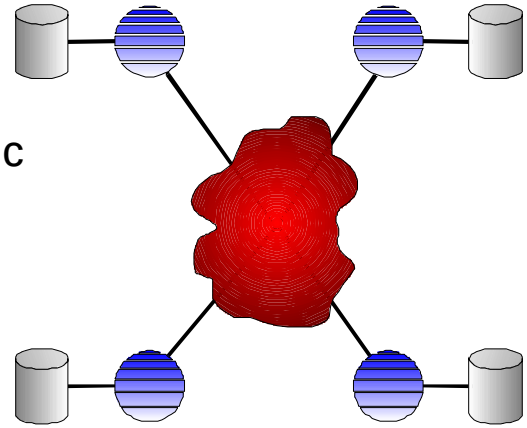
Storage Area Network

fibre channel, iSCSI



Cluster with dedicated I/O (block server) nodes

Symmetric cluster



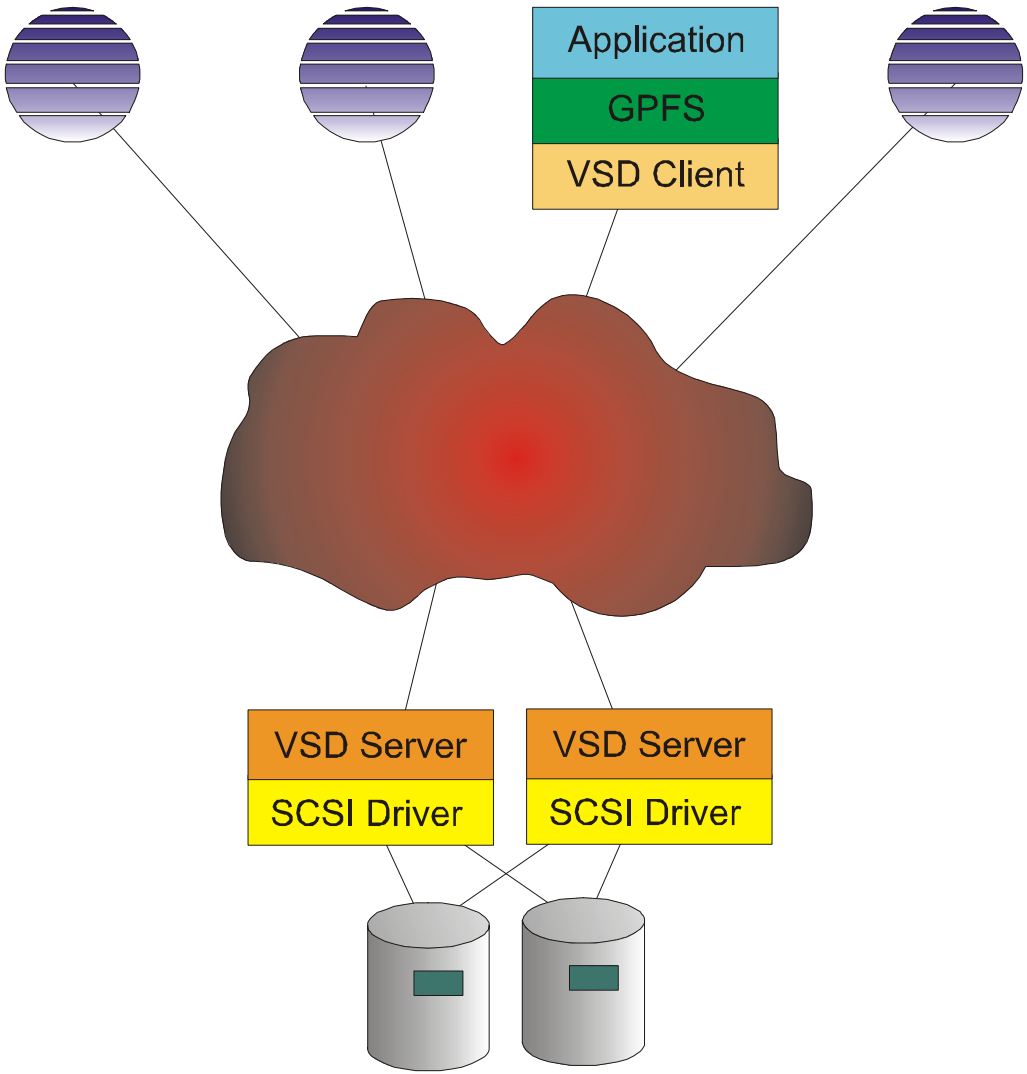
Software Shared Disk

NSD (GPFS internal)

IBM Confidential; subject to change without notice

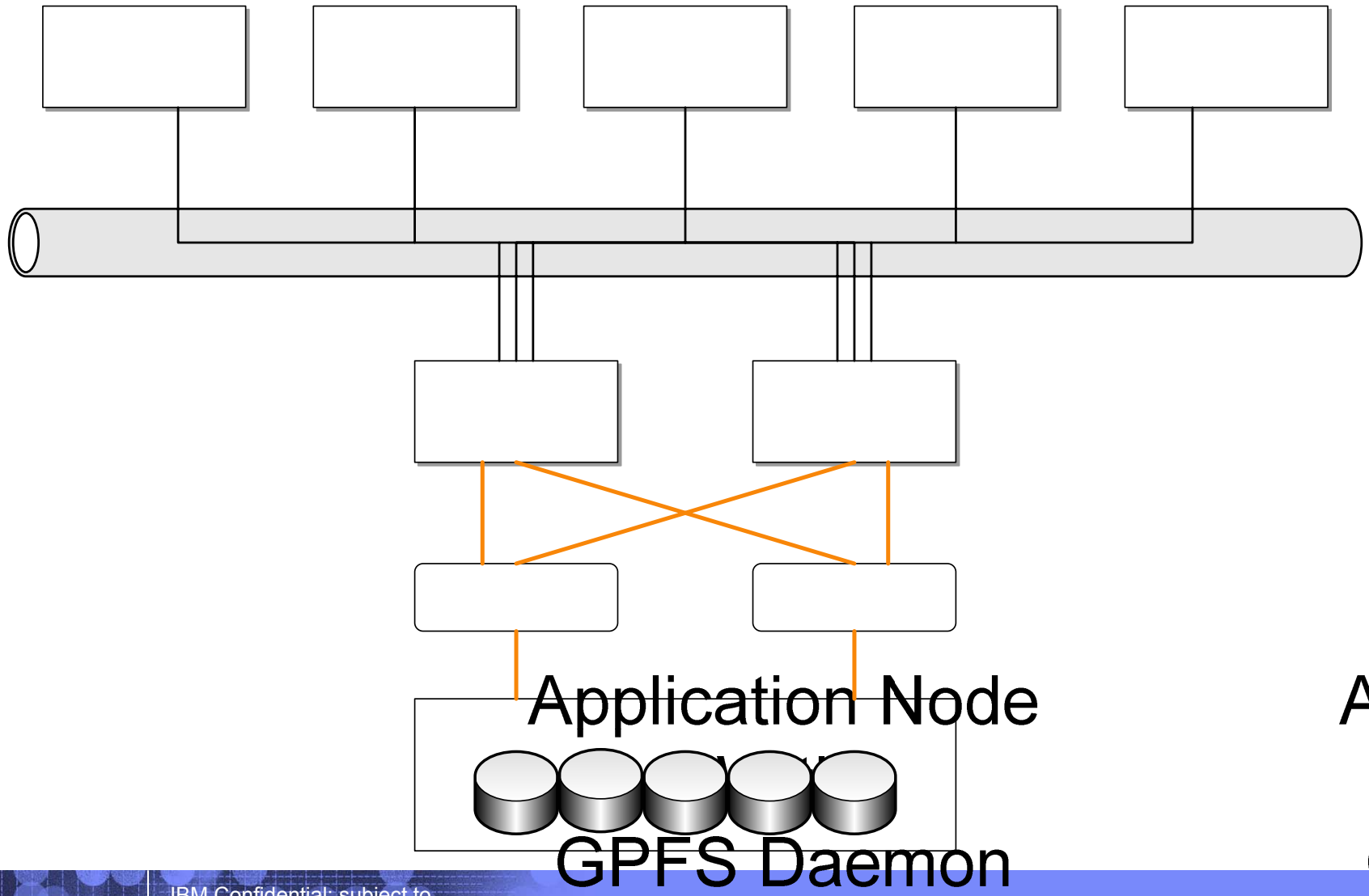
Simulated SAN: Virtual Shared Disk

- **VSD (Virtual Shared Disk) on SP**
  - Client is block device driver
  - Server is interrupt-level storage router
  - Highly AIX kernel dependent
- **NSD (network shared disk), part of GPFS, simulates SAN on Linux**
  - Data transported via GPFS RPC's
  - TCP transport on IP networks



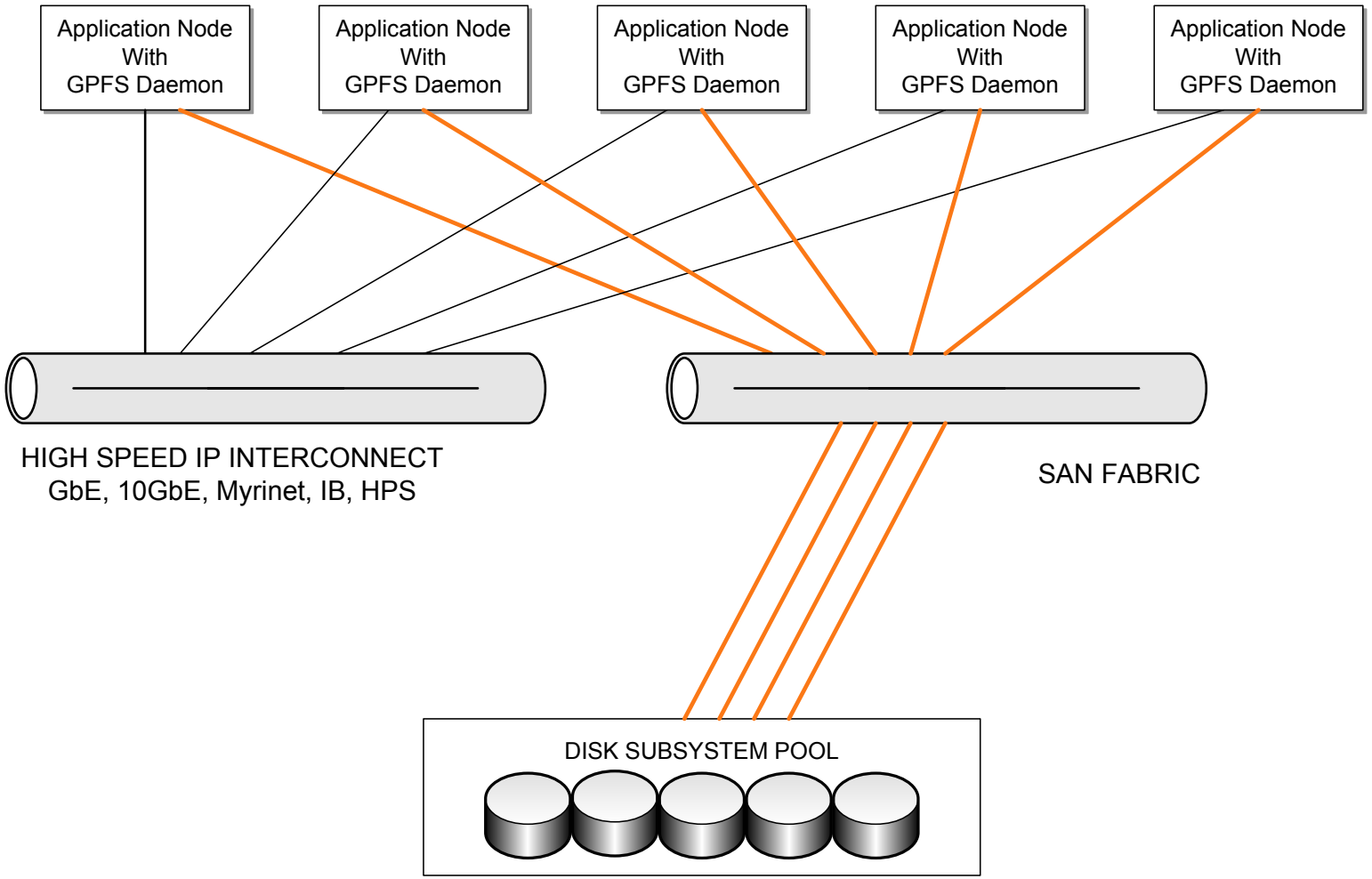
IBM Confidential; subject to change without notice

GPFS Model – VSD/NSD Servers



IBM Confidential; subject to change without notice

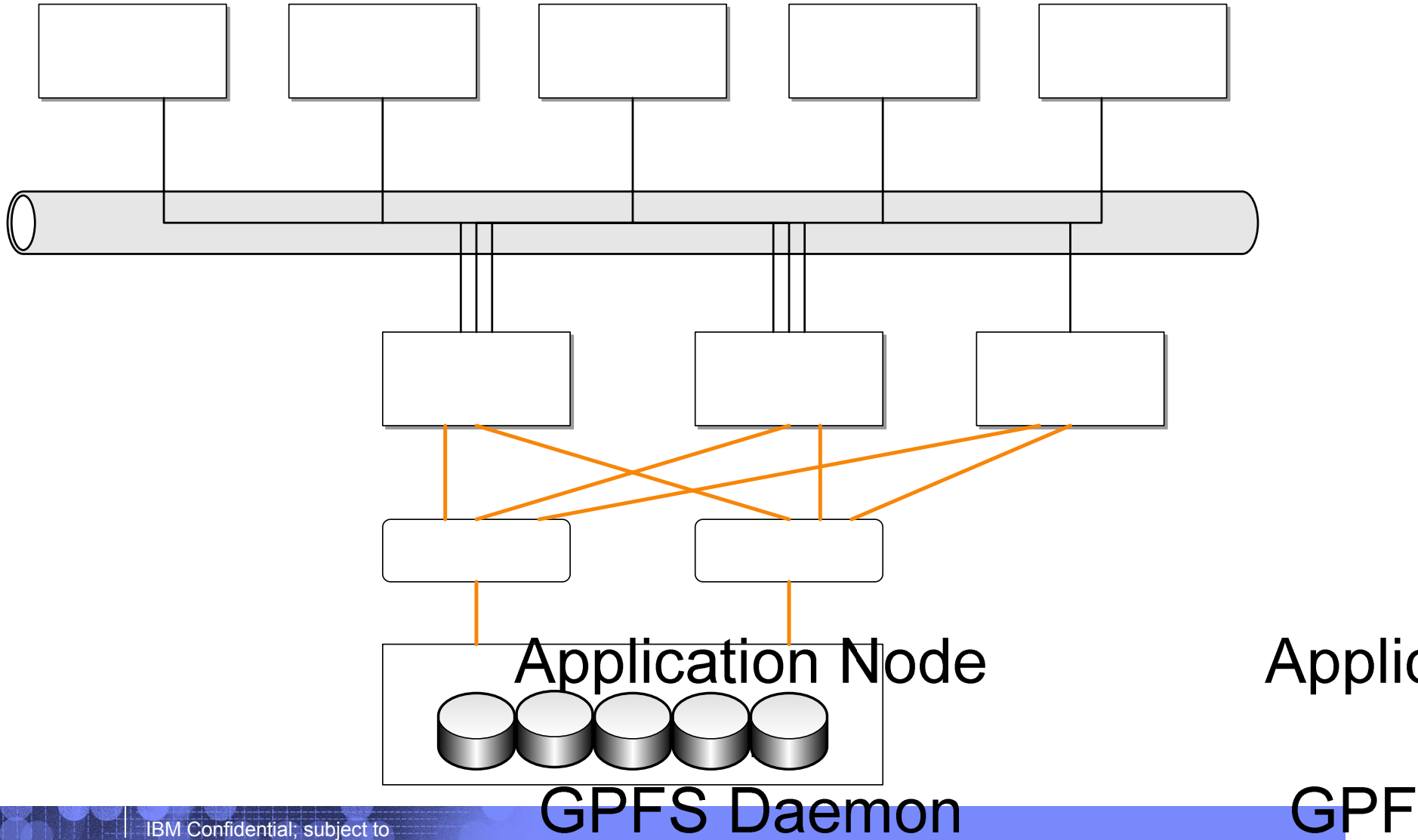
GPFS Model – SAN Attached Nodes



- **Each node has direct access to all disks and the IP network**

IBM Confidential; subject to change without notice

GPFS Model – NSD Servers/SAN Mixed



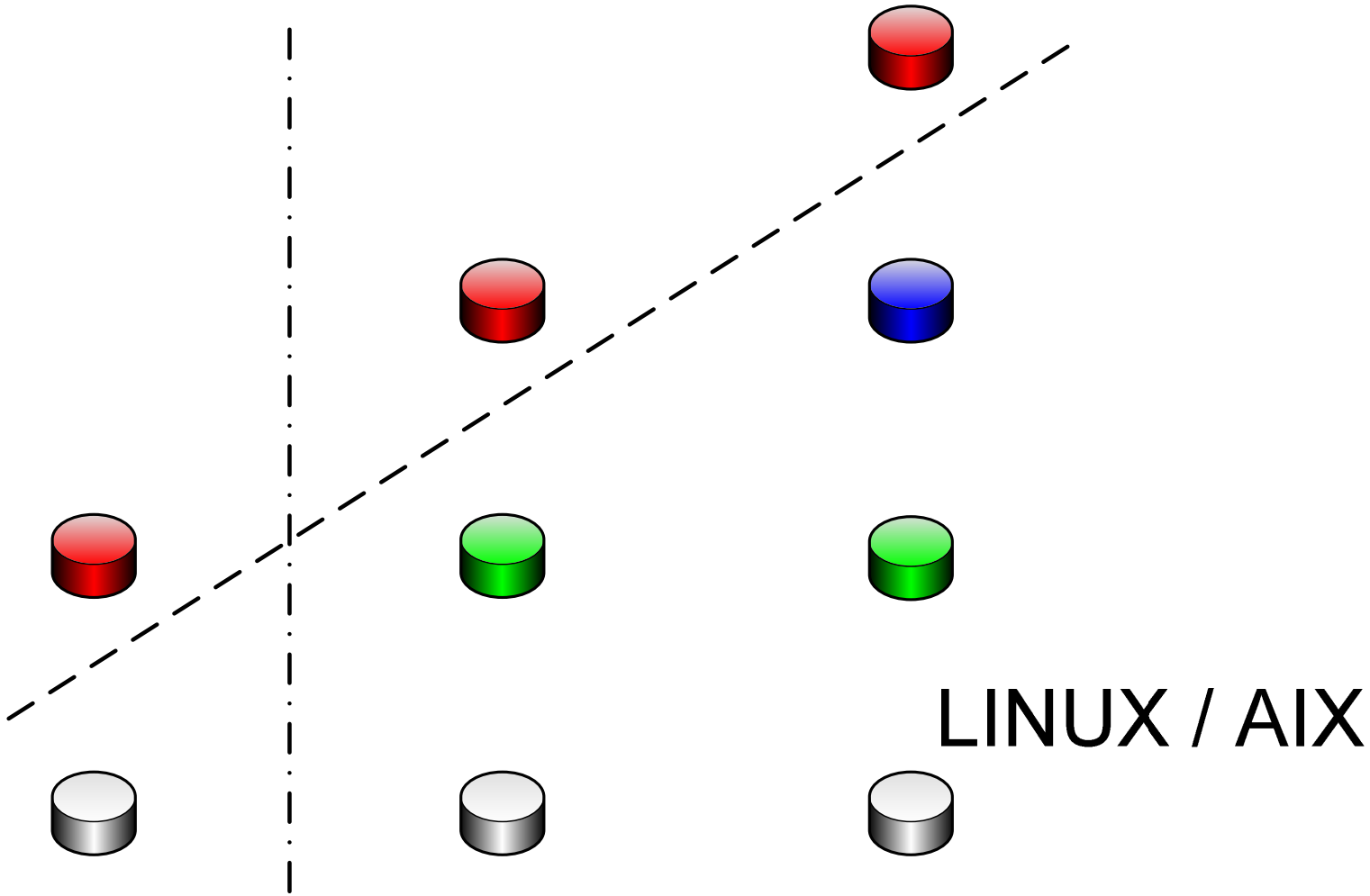
IBM Confidential; subject to change without notice

Configuration Options - NSD

- **Network Shared Disk**
  - A globally accessible name used to describe disks for use by GPFS in the GPFS cluster .
  - Disks are presented to GPFS as NSDs.
  - NSDs can be defined on various underlying device types.
    - /dev/sd*
    - VSDs
    - vpath
    - Emcpower

IBM Confidential; subject to change without notice

UNDERSTANDING THE NSD LAYER



IBM Confidential; subject to change without notice

Configuration Options - NSD

- **Network Shared Disk**
  - NSDs used in a NSD server configuration will have a primary server and optionally a secondary server identified.
  - For a higher degree of NSD availability it is recommended to define a primary and secondary server for each NSD even in the SAN model. The node can then revert to accessing the NSD(s) over the IP network in the event of a loss of SAN connectivity.

IBM Confidential; subject to change without notice

Configuration Options – Dynamic Discovery

- **NSD - What is dynamic discovery**
  - During GPFS initialization on a node the GPFS daemon will attempt to read all local disks searching for NSD disks. If a NSD is discovered locally, then the local path will be used. If the local path fails, then the network NSD server path will be used provided a primary and/or secondary server have been defined for the NSD.
  - Ability to control automatic discovery via –o useNSDserver option

IBM Confidential; subject to change without notice

Configuration Options – Dynamic Discovery

- **NSD – useNSDserver mount option**
  - On mmmount, mount, mmchfs, mmremotefs
    - -o useNSDserver={alwys | asfound | asneeded | never}
      - always: Local dynamic discovery is disabled; use NSD server
      - asfound: as found the first time disk was accessed
        - » No change from local to NSD server, or other way around
      - asneeded: Access the disk any way possible (This is the default)
      - never: Always use local disk access

IBM Confidential; subject to change without notice

GPFS Cluster Structure

- GPFS is a clustered file system defined over a number of nodes. This set of nodes is known as a "GPFS Cluster".
- Within one physical Cluster, there can exist one or more GPFS clusters.
  - Production cluster
  - Test cluster
- The nodes within each cluster share a set of file systems. These can be accessible by nodes in another cluster if enabled for Multi-Cluster mounts

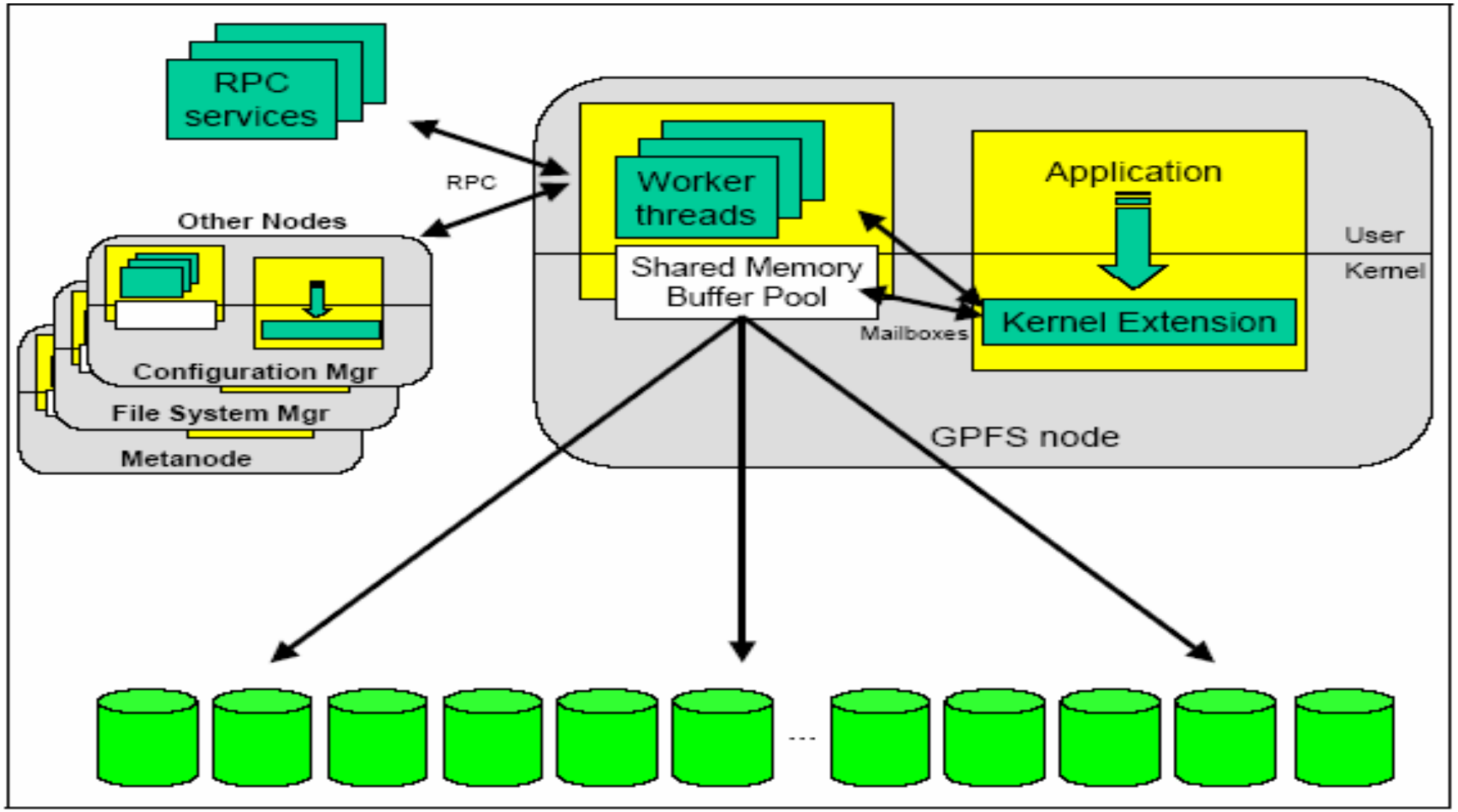
IBM Confidential; subject to change without notice

GPFS Node Structure

- **On each node in the cluster, GPFS consists of:**
  - Administrative Commands
  - A kernel extension
  - A multi-threaded daemon
  - Linux nodes within cluster will have the GPFS open source portability layer
    - The portability layer is unique for each architecture/distribution/kernel combination.

IBM Confidential; subject to change without notice

GPFS Node Architecture



IBM Confidential; subject to change without notice

GPFS Administrative Commands

- /usr/lpp/mmfs/bin
  - Limited number executable by non-root user
    - mmbackup
    - mmdf
    - mmlsdisk
    - mmlsfs
    - mmlsmgr
    - mmlsquota
    - mmsnapdir
    - mmsnaplatest
- All commands have prefix “mm”
  - mmlscluster
  - mmlsconfig …
- ksh shell scripts

IBM Confidential; subject to change without notice

GPFS Open Source Portability Layer (GPL)

- (ONLY) Linux nodes running GPFS.
- mmfslinux portability module
- Enables communication between GPFS kernel modules and the Linux kernel
- Must be built during installation: directions are in the file /usr/lpp/mmfs/src/README
- The portability layer is unique for each architecture/distribution/kernel combination.

IBM Confidential; subject to change without notice

GPFS Kernel Extension

- The GPFS kernel extension provides:
  - Interfaces to the operating system vnode and VFS
- Flow:
  - Application makes filesystem calls to the O/S
  - O/S presents calls to the GPFS kernel extension
    - Net: GPFS appears to the application as just another filesystem
  - GPFS kernel extension will either satisfy requests using resources already available, or, send a message to the GPFS daemon to complete the request

IBM Confidential; subject to change without notice

GPFS Daemon

- The GPFS daemon performs all I/O and buffer management, including read ahead for sequential reads and write behind for all writes not specified as synchronous.
- All I/O is protected by token management to ensure file system consistency.
- Multi-threaded with some threads dedicated to specific functions.
  - examples include allocation, directory management (insert and removal), quotas
- Disk I/O is initiated on threads of the daemon

IBM Confidential; subject to change without notice

Configuration Options – OS Choices

- Clusters can be made up of AIX, Linux, or both AIX and Linux nodes.
- Linux nodes can be of various platform types within a cluster (pLinux, xLinux IA32, EM64T and blades)
- Various Linux distros and kernels are supported in different combinations, refer to the GPFS FAQ Page for details.

IBM Confidential; subject to change without notice

GPFS Version 3.1 Supported On:

	SLES9	SLES10	RH4	RH5	AIX 5.2	AIX 5.3
Power	YES	YES	YES	Coming	RPQ	YES
x86	YES	YES	YES	Coming	N/A	N/A
IA64	Special Bid	?	Not planned	?	N/A	N/A
EM64T (64 bit)	YES	YES	YES	Coming	N/A	N/A

IBM Confidential; subject to change without notice

GPFS Cluster Structure

- GPFS is a clustered file system defined over a number of nodes. This set of nodes is known as a "GPFS Cluster".
- Within one physical Cluster, there can exist one or more GPFS clusters.
  - Production cluster
  - Test cluster
- The nodes within each cluster share a set of file systems. These can be accessible by nodes in another cluster if enabled for Multi-Cluster mounts

IBM Confidential; subject to change without notice

GPFS Node Structure

- **On each node in the cluster, GPFS consists of:**
  - Administrative Commands
  - A kernel extension
  - A multi-threaded daemon
  - Linux nodes within cluster will have the GPFS open source portability layer
    - The portability layer is unique for each architecture/distribution/kernel combination.

IBM Confidential; subject to change without notice

Special Nodes

Cluster Configuration Servers (Primary/Secondary):

- **Hold master/backup copy of static configuration data**

Configuration manager [-> Cluster Manager]

- **Monitors node status and detects failures**
- **Grants disk leases**
- **Drives node failure recovery**
- **Appoints File System managers**
- **Elected from a designated set of "core nodes"**

File system manager:

- **One per file system**
- **Runs log recovery**
- **creates & assigns log files**
- **Acts as a token server (version 3.1 update: multiple token managers)**
- **Acts as allocation manager**
- **Quota management in a quota-enabled filesystem**
- **Runs/coordinates file system management operations (crfs, delfs, adddisk, deldisk, restripe, repair, defrag, …)**

IBM Confidential; subject to change without notice

Special Nodes (cont)

Token Manager

- **Manage portion of tokens for each filesystem**

Helper nodes

- **Enlisted by FS manager to help moving/copying data**

Metanode

- **One per file, elected dynamically**
- **Performs all metadata I/O for a file (inode, indirect blocks)**
- **Collects/merges metadata updates from other nodes**

Remote nodes

- **Node from a separately administered GPFS cluster**
- **Cannot become FS manager or helper node, but can become metanode**

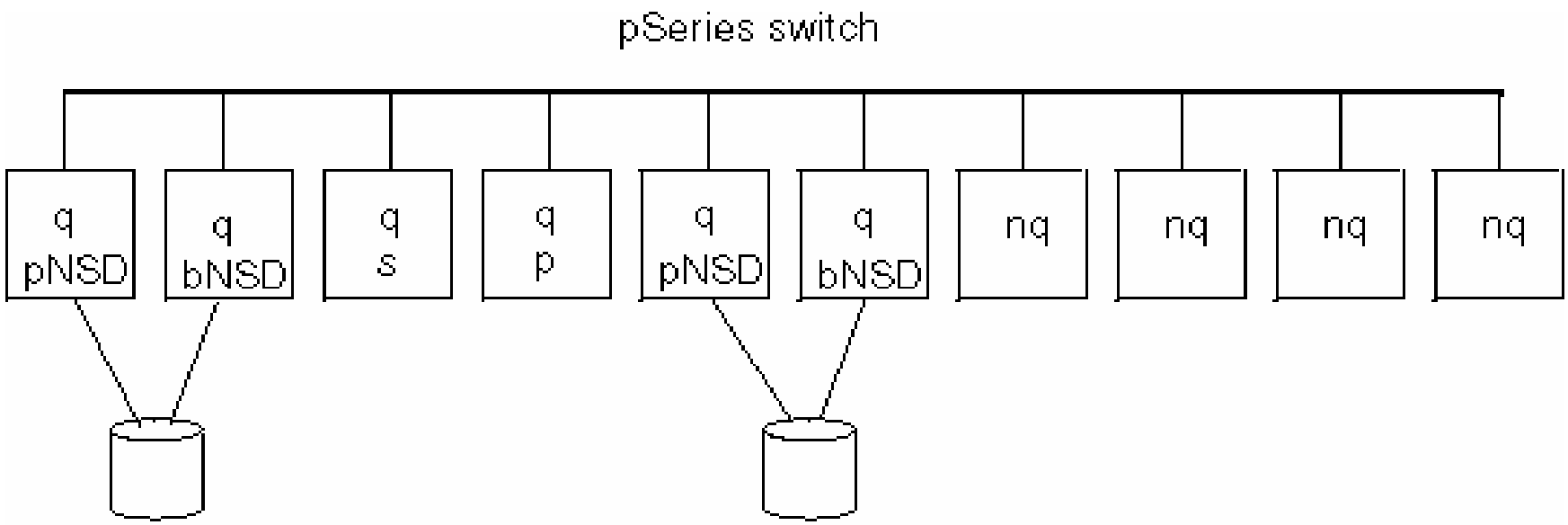
IBM Confidential; subject to change without notice

Configuration Options – Quorum Types

- **Node Quorum**
  - Only a subset of nodes are designated as quorum nodes. This speeds up cluster formation as well as allows any number of non-quorum nodes from failing without impacting the overall quorum calculation.
  - Node quorum is defined as one plus half of the explicitly defined quorum nodes in the GPFS cluster
  - This is the default quorum algorithm
    - mmfsadm quorum <new_quorum_value>
      - only when directed by IBM Service

IBM Confidential; subject to change without notice

Node Quorum Example



p - primary cluster configuration server

s - secondary cluster configuration server

q - quorum node

nq - non-quorum node

pNSD - primary NSD server

bNSD - backup NSD server

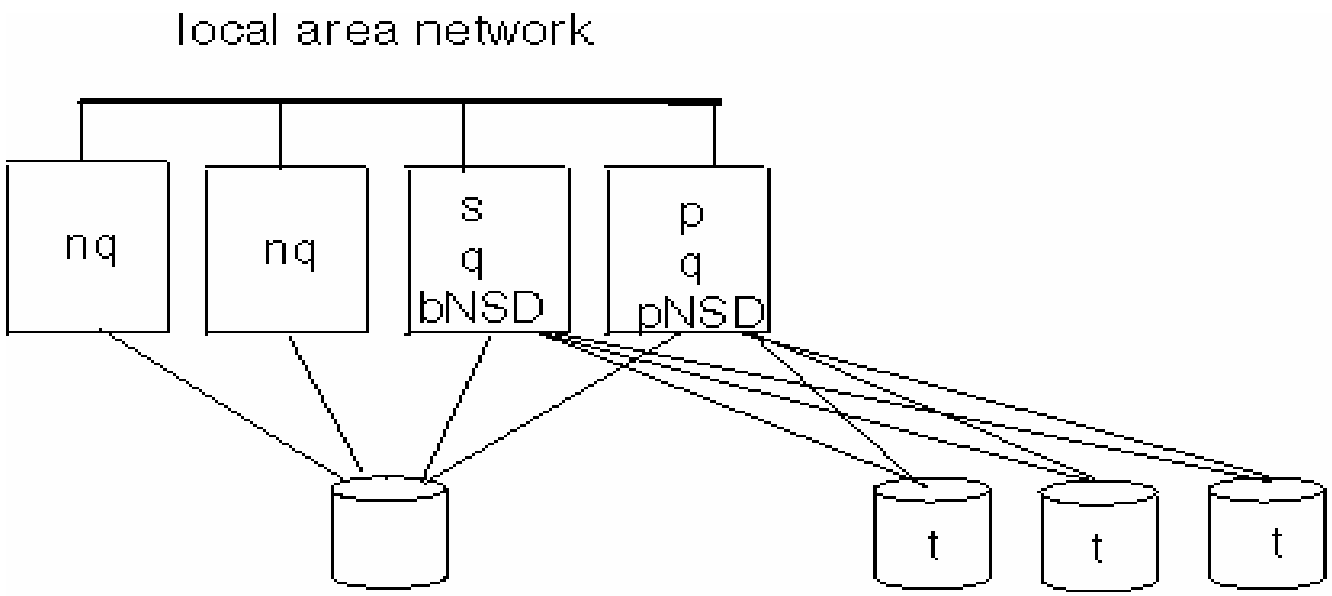
IBM Confidential; subject to change without notice

Configuration Options – Quorum Types

- **Node Quorum with Tiebreaker Disks**
  - Used in configs with TWO quorum nodes [version 2.3]
    - Up to 8 quorum nodes [version 3.1]
  - In addition to quorum nodes, between One and Three disks (NSDs) are used as Tiebreaker disks
    - Odd number of disks is recommended (1, 3)
    - Disks (NSDs) must be accessible from the quorum nodes
    - Need not belong to any particular filesystem
  - Recommended that these quorum nodes are also the primary and secondary GPFS configuration servers
  - Use the tiebreakerDisks parameter on the mmchconfig command to enable this quorum mechanism
    - mmchconfig tiebreakerDisks="gpfs1nsd;gpfs2nsd;gpfs3nsd"

IBM Confidential; subject to change without notice

Node Quorum with Tiebreaker Disks Example



p - primary cluster configuration server
s - secondary cluster configuration server
q - quorum node
nq - non-quorum node

pNSD - primary NSD server
bNSD - backup NSD server
t - tiebreaker disk

IBM Confidential; subject to change without notice

Quorum (1)

- **mmlscluster**

```
GPFS cluster information
========================
  GPFS cluster name:         rr_beta.ppd.pok.ibm.com
  GPFS cluster id:           13882466803121521035
  GPFS UID domain:           rich_domain
  Remote shell command:      /usr/bin/rsh
  Remote file copy command:  /usr/bin/rcp

GPFS cluster configuration servers:
-----------------------------------
  Primary server:    c35f1n01.ppd.pok.ibm.com
  Secondary server:  c35f1n03.ppd.pok.ibm.com

 Node  Daemon node name            IP address       Admin node name             Designation
-----------------------------------------------------------------------------------------------
   1   c35f1n01.ppd.pok.ibm.com    192.168.110.1    c35f1n01.ppd.pok.ibm.com    quorum-manager
   2   c35f1n02.ppd.pok.ibm.com    192.168.110.2    c35f1n02.ppd.pok.ibm.com    quorum-manager
   3   c35f1n03.ppd.pok.ibm.com    192.168.110.3    c35f1n03.ppd.pok.ibm.com    quorum-manager
   4   c13sun16g.ppd.pok.ibm.com   192.168.120.16   c13sun16g.ppd.pok.ibm.com
   5   c13sun19g.ppd.pok.ibm.com   192.168.120.19   c13sun19g.ppd.pok.ibm.com
   6   c13sun17g.ppd.pok.ibm.com   192.168.120.17   c13sun17g.ppd.pok.ibm.com
   7   c13sun18g.ppd.pok.ibm.com   192.168.120.18   c13sun18g.ppd.pok.ibm.com
```


IBM Confidential; subject to change without notice

Quorum (2)

- **root@c5n50:/root # mmgetstate**

```
 Node number  Node name        GPFS state
------------------------------------------
       3      c5sn50           active
```

- **root@c5n50:/root # mmgetstate -aLs**

```
 Node number  Node name       Quorum  Nodes up  Total nodes  GPFS state  Remarks
-----------------------------------------------------------------------------------
       1      c5sn48             2        3          3       active      quorum node
       2      c5sn49             2        3          3       active      quorum node
       3      c5sn50             2        3          3       active      quorum node

 Summary information
---------------------
Number of nodes defined in the cluster:            3
Number of local nodes active in the cluster:       3
Number of remote nodes joined in this cluster:     0
Number of quorum nodes defined in the cluster:     3
Number of quorum nodes active in the cluster:      3
Quorum = 2, Quorum achieved
```


IBM Confidential; subject to change without notice

Quorum (3)

- `c35f1n01:~ # mmfsadm dump cfgmgr`

```
nClusters 1

Cluster Configuration [0] "rr_beta.ppd.pok.ibm.com": Type: 'LC' id C0A86E014636018B
ccUseCount 0 unused since (never) contactListRefreshMethod 0
Domain .ppd.pok.ibm.com, myAddr 0 192.168.110.1, authIsRequired false
UID domain 0x10003B7ED0 (0xFFFFFF00003B7ED0)  Name "rich_domain"
  hold count 1 CredCacheHT 0x0 IDCacheHT 0x0
No of nodes: 7 total, 7 local, 3 quorum nodes.

Authorized keys list:
clusterName               port cKeyGen nKeyGen cipherList

Cluster info list:
clusterName               port cKeyGen nKeyGen cipherList
rr_beta.ppd.pok.ibm.com  1191      -1      -1 EMPTY

node                      primary      admin --status---   join fail  SGs -lease-renewals-- --heartbeats-- other ip addrs,
  no   idx host name      ip address    func tr p    rpc  seqNo  cnt mngd sent    processed received pings last failure
---- ----- -------------- ------------ ----- ----------- ------ ---- ---- ---------------- -------------- -------------------
   1     0 c35f1n01       192.168.110.1  qm--l -- J     up      1    0   0     0           40237.72 40237.72
   2     1 c35f1n02       192.168.110.2  qm--l -- J     up      1    0   0     0 40295.71 40295.95 40295.95
   3     2 c35f1n03       192.168.110.3  qm--l -- J     up      1    0   0     0 40296.73 40297.44 40297.44
   4 65539 c13sun16g      192.168.120.16 ----l -- -   down      0   -1   0     0
   6 65538 c13sun17g      192.168.120.17 ----l -- -   down      0   -1   0     0
   7 65537 c13sun18g      192.168.120.18 ----l -- -   down      0   -1   0     0
   5 65536 c13sun19g      192.168.120.19 ----l -- -   down      0   -1   0     0
```


IBM Confidential; subject to change without notice

Quorum (4): mmfsadm dump cfgmgr continued

```
Current clock tick (seconds since boot): 1940304.23 (resolution 0.010) = 2007-05-06 20:33:13

Groupleader 192.168.110.1 0x00000000 (this node)
Cluster configuration manager is 192.168.110.1 (this node); pendingOps 0
group quorum formation time 2007-05-06 20:32:06
gid 463e7386:c0a86e01 elect <1:174> seq 1 pendingSeq 1, gpdIdle phase 0, joined
GroupLeader: joined 1 gid <1:174>
useDiskLease yes, leaseDuration 35 recoveryWait 35 dmsTimeout 23
lastFailedLeaseGranted: 0.00
lastLeaseObtained      1940296.73, 27.50 sec left (ok)
leaseQuorum 2
renewalList 1 of 1: (2 40296.73)
ping timeouts: period 2 max missed 17 max sent 60
stats: nTemporaryLeaseLoss 0 nSupendIO 0 nLeaseOverdue 0 nTakeover 0 nPinging 0
Summary of lease renewal round-trip times:
  Number of keys = 2, total count 4
  Min 0.0 Max 0.5, Most common 0.0 (3)
  Mean 0.1, Median 0.0, 99th 0.5
ccSANergyExport no

Assigned stripe group managers:
  "sunalpha"  id 00000000:00000000  cond 0x409C2090  sgiHold 1  SG mgr not appointed seq 0
  None appointed; done recovery=true
    DMAPI disposition is not set for any event.
  sgStats: tmSpace 0 freeSpace 0 tmRequests 0 cpuUsage 0 totalTmRequests 0
```


IBM Confidential; subject to change without notice

NSD: Replication /Failure Groups and Storage Pools

- **Creation of NSD requires [mmcrnsd]**
  - O/S disk name
  - Primary / Backup Server nodes
    - Optional, but recommended
  - NSD name
  - Failure Group (related to Replication)
  - Storage Pool (related to Policy / ILM)
- **Disk Descriptor**
  - example:
    - /dev/sdb:c35f1n01:c35f1n02::1:nsdb1_1:system

IBM Confidential; subject to change without notice

NSD: List the NSDs

- **mmlsnsd**

```
 File system   Disk name    NSD servers
---------------------------------------------------------------------------
 sunalpha      nsdb1_1      c35f1n01.ppd.pok.ibm.com,c35f1n02.ppd.pok.ibm.com
 sunalpha      nsdb1_2      c35f1n01.ppd.pok.ibm.com,c35f1n02.ppd.pok.ibm.com
 sunalpha      nsdb1_3      c35f1n01.ppd.pok.ibm.com,c35f1n02.ppd.pok.ibm.com
 sunalpha      nsdb1_4      c35f1n01.ppd.pok.ibm.com,c35f1n02.ppd.pok.ibm.com
```


IBM Confidential; subject to
change without notice

GPFS: Tunables

- **pagepool size(s)**
- **maxFilesToCache**
- **maxStatCache**
- **autoload**
- **healthCheckInterval**

IBM Confidential; subject to change without notice

Cache Usage

- **GPFS creates a number of cache segments**
  - Controlled by 3 attributes
    - Attributes have default values at cluster creation
    - May be altered via mmchconfig
      - (-N): changeable on node by node basis
      - (-i) : change takes effect, immediately, and persists
      - (-I) : change takes effect, immediately, does not persist

IBM Confidential; subject to change without notice

pagepool

- **Caches user file data and file system metadata**
  - Enable asynchronous I/O
- **Minimum: 4 MB**
- **Default: 64 MB**
- **Maximum: Dependent on amount of physical memory and operating system**
  - No greater than 50% of real memory
- **Can use “-i” or “-I” and “-N” on mmchconfig**

IBM Confidential; subject to change without notice

maxFilesToCache

- **Number of inodes to cache for recently used files that have been closed.**
- **Provides faster re-access to the file**
- **Minimum: 1**
- **Default: 1000**
- **Maximum: 100 000**
- **Recommendation: large enough to handle number of concurrently open files + allow caching of recently used files**
- **Can use "-N" on mmchconfig**
- **Total memory required for inodes + control data**
  - maxFilesToCache x [2 KB = 512 byes for an inode]
- **Particularly useful for systems accessing a large number of small files**

IBM Confidential; subject to change without notice

maxStatCache

- **Additional memory to cache attributes not currently in the regular file cache**
  - Useful to improve system and GPFS stat() calls
  - Memory used: maxStatCache x 176 bytes
- **Number of inodes to keep in stat cache**
- **Provides only enough information to perform query on filesystem (e.g., stat())**
- **Minimum: 0**
- **Default: 4 x maxFilesToCache**
- **Maximum: 10 000 000**
- **Can use "-N" on mmchconfig**
- **Benefitors: applications that test existence of file and/or properties of files without actually opening file, e.g., backup application**

IBM Confidential; subject to change without notice

autoload

- **Start (load) GPFS whenever node is rebooted**
- **Valid values (yes | no)**

IBM Confidential; subject to
change without notice

distributedTokenServer

- **Specifies whether token server role is**
  - Limited only to FileSystem manger node (no)
  - Distributed to other nodes (yes) : Default
    - Improve filesystem performance
      - Selected from among nodes designated as “manager”

IBM Confidential; subject to change without notice

healthCheckInterval

- Improve GPFS' ability to detect "bad" nodes
  - ping ok, lease renewals ok, BUT can't do "other" things
  - Two known cases:
    - One of the mmfsd threads is killed by the Linux page fault handler
      - Will see "VM: killing process mmfsd" message in syslog
    - File system NFS mounted and NFS server is unavailable/hung
- New thread, Healthcheck Watchdog thread is created during GPFS daemon startup IF healthCheckInterval is non-zero
  - the daemon has to be restarted in order for a changed healthCheckInterval value to become effective.
- Specifies number of seconds between health checks
  - Default: 0
  - Minimum: 10
  - Maximum: 7200

IBM Confidential; subject to change without notice

GPFS Releases

- **Recent Releases**
  - GPFS 2.3 shipped 12/04
  - GPFS 3.1 shipped 4/06
- **Planned**
  - GPFS 3.2 planned for 8/07 GA
  - DARPA High Productivity Computing Systems program (HPCS) deliveries through 2010

IBM Confidential; subject to change without notice

GPFS 2.3 (Dec 04)

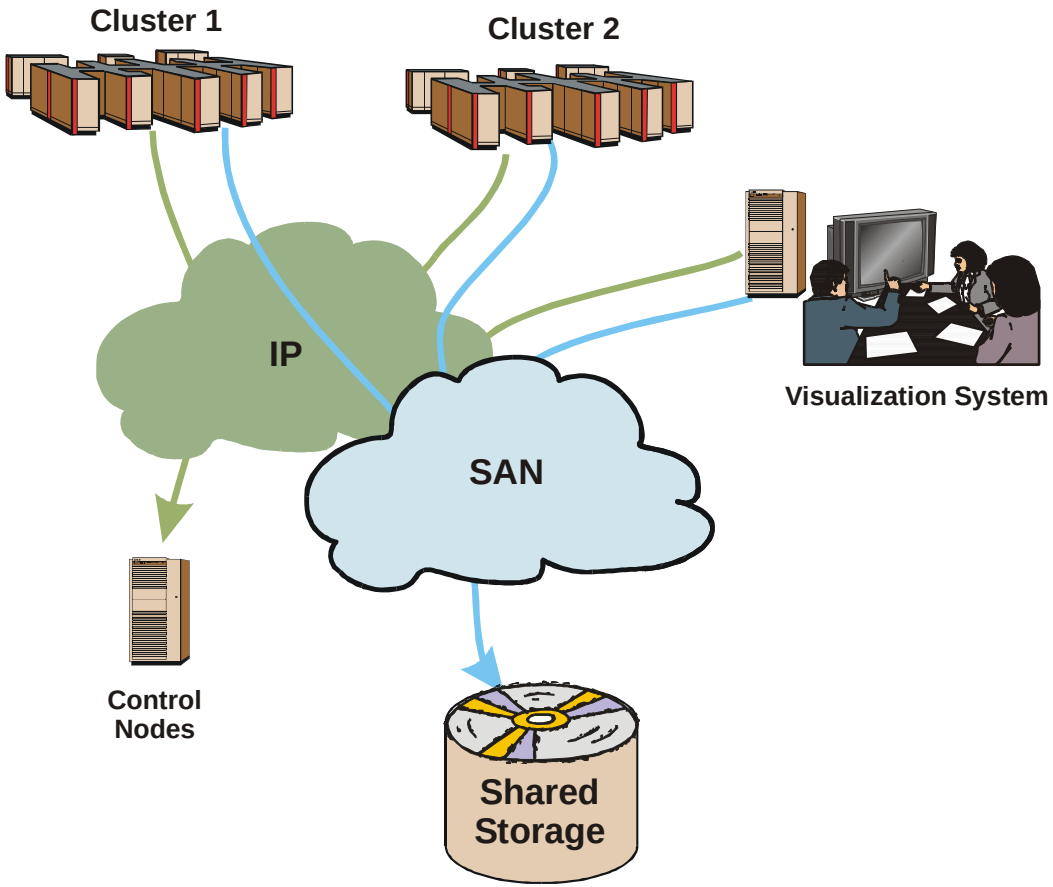
- **Linux 2.6 kernel**
  - SLES 9/RH EL 4
- **Larger File System / disk address support**
  - AIX only right now due to OS and device driver limits in Linux
- **NFSv4 (AIX)**
- **Larger clusters**
  - Up to 2400 installed
- **WAN / multi-cluster**
  - Multiple sites can cross-mount GPFS file systems over sufficiently fast WAN links (Grid projects)
  - Linking computing resources within or across organizations:
    - Run applications anywhere
    - Use otherwise idle resources
    - Enable collaboration, e.g.:

      collect->process->visualize

IBM Confidential; subject to change without notice

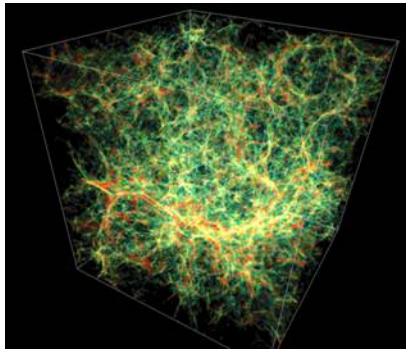
GPFS beyond the Cluster

- **Problem: In a large data center, multiple clusters and other nodes need to share data over the SAN**
- **Solution: eliminate the notion of a fixed cluster**
  - "control" nodes for admin, managing locking, recovery, …
  - File access from "client" nodes
    - Client nodes authenticate to control nodes to mount a file system
    - Client nodes are trusted to enforce access control
    - Clients still directly access disk data and metadata

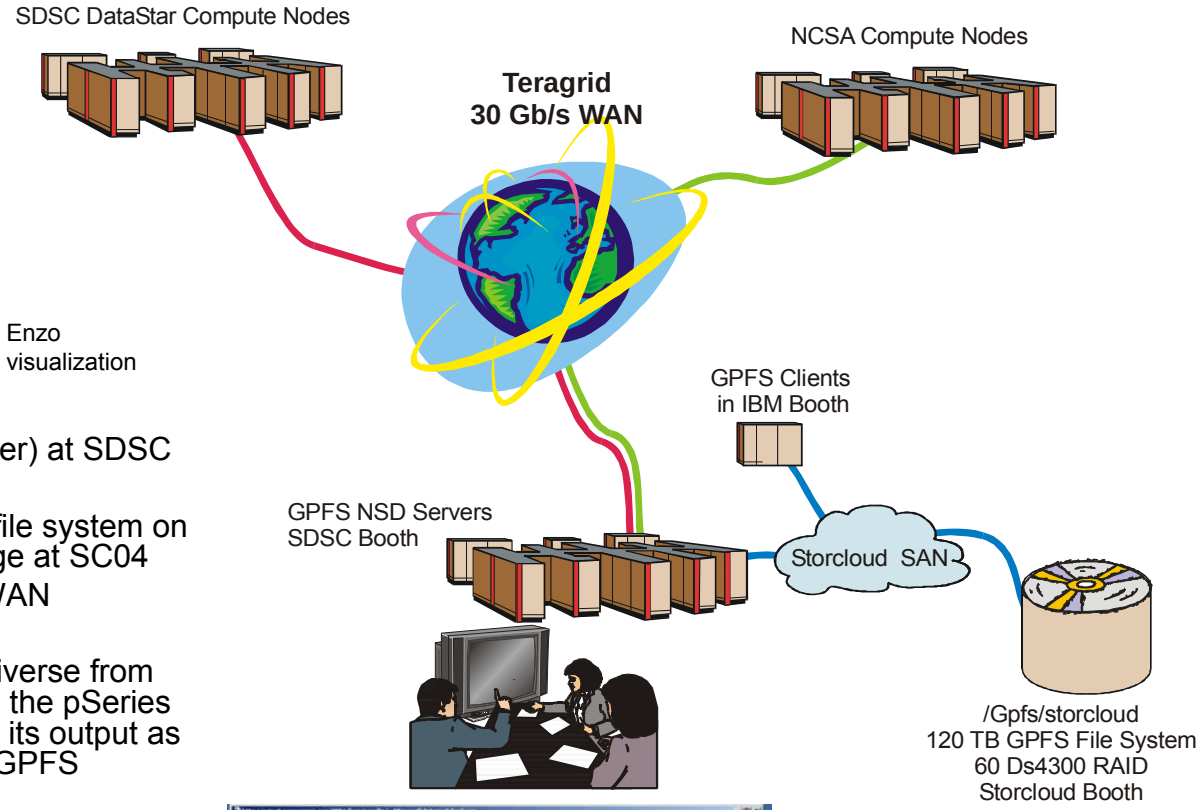


IBM Confidential; subject to change without notice

SDSC-IBM StorCloud Challenge Demo – Grid Storage



Enzo visualization

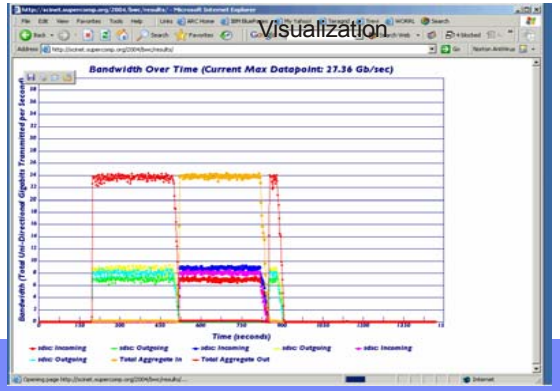


Grid Sites

- Data Star (183-node IBM pSeries cluster) at SDSC
- Mercury IBM IA64 cluster at NCSA
- GPFS 2.3 (Multi-cluster) 120-terabyte file system on IBM DS4300 (FAStT 600 Turbo) storage at SC04
- Teragrid 30 Gb/s Teragrid backbone WAN

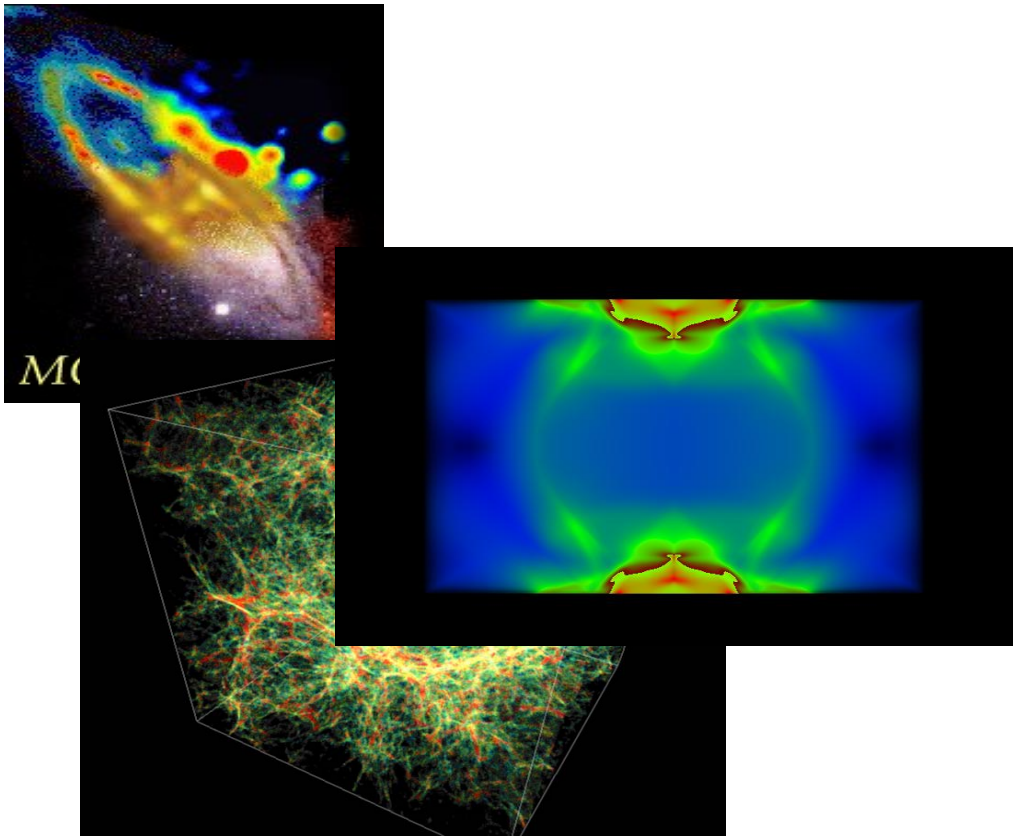
Workflow

- **Enzo** simulates the evolution of the universe from big bang to present. Enzo runs best on the pSeries Data Star nodes at SDSC. Enzo writes its output as it is produced over the Teragrid to the GPFS Storcloud file system at SC04.
- **VISTA** reads the Enzo data from the GPFS StorCloud file system and renders it into images that can be compressed into a QuickTime video. Vista takes advantage of the cheaper IA64 nodes at NCSA. VISTA reads the Enzo results from the StorCloud file system and writes its output images there as well.
- The resulting QuickTime movie is read from GPFS and displayed in the SDSC booth at the conference



IBM Confidential; subject to change without notice

Grid Computing

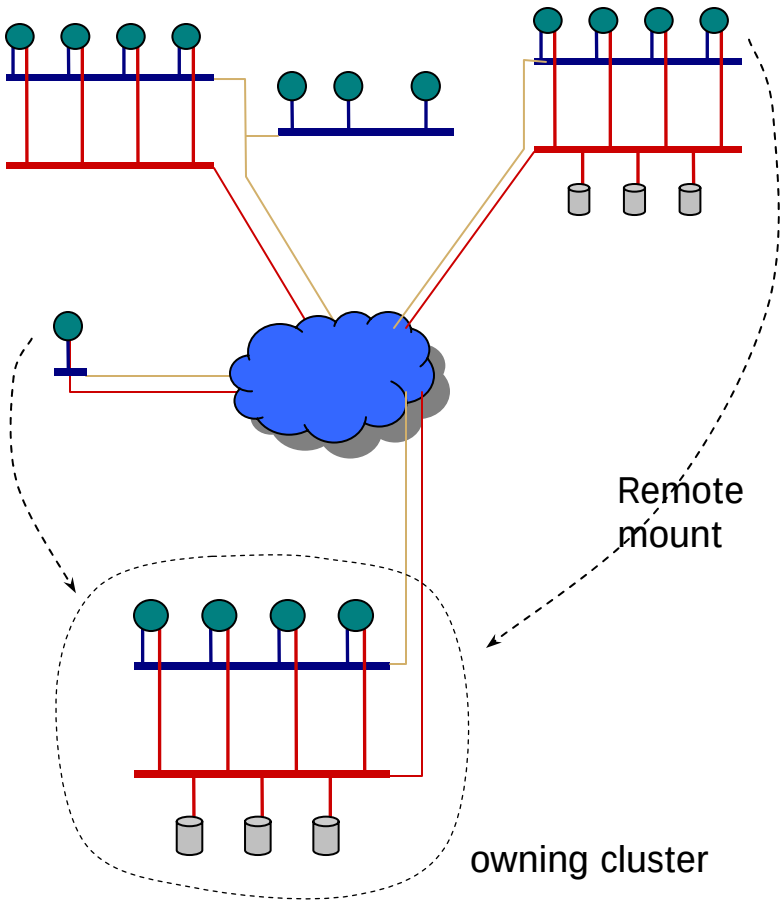


- **Dispersed resources connected via a network**
  - Compute
  - Visualization
  - Data
  - Instruments (telescopes, microscopes, etc.)
- **Sample Workflow:**
  - Ingest data at Site A
  - Compute at Site B
  - Visualize output data at Site C
- **Logistic nightmare!**
  - Conventional approach: copy data via ftp. Space, time, bookkeeping.
  - Possible approach: on-demand parallel file access over grid
    - … but the scale of problems demand high performance!

IBM Confidential; subject to change without notice

GPFS Cross-Cluster Mounts

- **Each FS belongs to one "owning" cluster, responsible for**
  - Administration, lock management and recovery
- **"Remote" nodes can mount FS and request locks:**
  - Access data & metadata directly over the SAN
- **Security capability exist where**
  - Nodes authenticate themselves to the owning cluster.
- **User IDs in one cluster can be mapped to user IDs in the owning cluster**
- **Performance largely a function of the network**



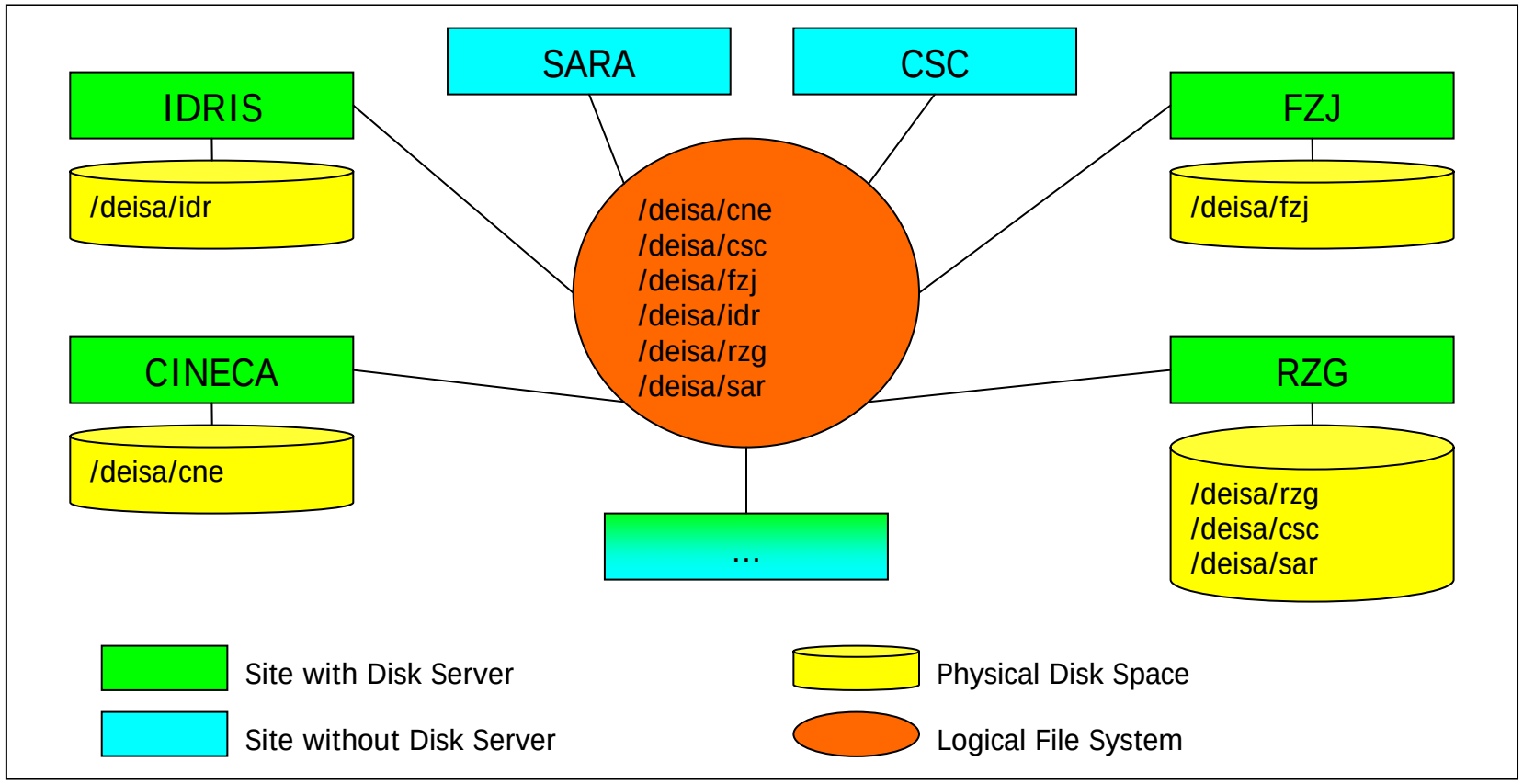
IBM Confidential; subject to change without notice

Multi-cluster GPFS and Grid Storage

- **Multi-cluster**
  - Mount filesystem a file system owned and housed elsewhere
  - Parallel access via WAN connections
- **Multi-cluster works within a site or across a WAN**
  - Lawrence Berkeley Labs (NERSC)
    - multiple supercomputer clusters share large GPFS file systems
  - DEISA (European computing grid)
    - RZG, CINECA, IDRIS, CSC, CSCS, UPC, IBM
    - pSeries and other clusters interconnected with multi-gigabit WAN
    - Multi-cluster GPFS in “friendly-user” production 4/2005
  - Teragrid
    - **SDSC, NCSA, ANL, PSC, CalTech**, IU, UT, …..
    - Sites linked via 30 Gb/sec dedicated WAN
    - 500TB GPFS file system at SDSC shared across sites, 1500 nodes
    - In production, 4TB of new data each day
    - SDSC file system mounted by DEISA client nodes

IBM Confidential; subject to change without notice

DEISA use of GPFS Multi-cluster



IBM Confidential; subject to change without notice

GPFS Multicluster Enhancements

- **Basic multicluster functions in GPFS 2.3 (12/2004)**
  - Ability to define mount relationships (mmauth, mmremotecluster, mmremotefs)
  - Security levels via SSL connections (none, authentication, integrity, encryption)
  - User ID mapping to support multiple administrative domains
- **Improvements in 3.1**
  - No communication among client nodes – only between a client node and the server cluster it is mounting from
  - Token manager scaling – split token manager across several nodes to improve scaling (more nodes => more tokens)
  - Dynamic metanode selection
    - Metanode in client cluster if all accesses from there
    - Metanode in server cluster if file is shared across clusters

IBM Confidential; subject to change without notice

GPFS release 3.1 GA'ed Apr'06

- **Information lifecycle management in GPFS**
  - Filesets, storage pools and policies
- **Multi-clustering enhancements**
  - Finer-grained security (per filesystem, per cluster)
  - No peer-peer communication (universal connectivity) among client clusters
    - Adaptive selection of metanode
  - Multiple Network support – can use separate subnets for intra-cluster, extra-cluster, and admin communications (e.g. Infiniband and Ethernet)
  - Higher performance in security functions through improved threading
  - Release coexistence from 3.1 to later releases - helps (multi-cluster) migration.
- **Distributed token server**
  - Multiple token servers per filesystem
- **NFSV4 enhancements on AIX**
- **Various performance, function, and manageability features**
  - Cluster-wide mount/unmount utilities
  - Failback control after NSD failover
  - Autoconfig tool for Linux installations
  - Quotactl & fileset quotas

IBM Confidential; subject to change without notice

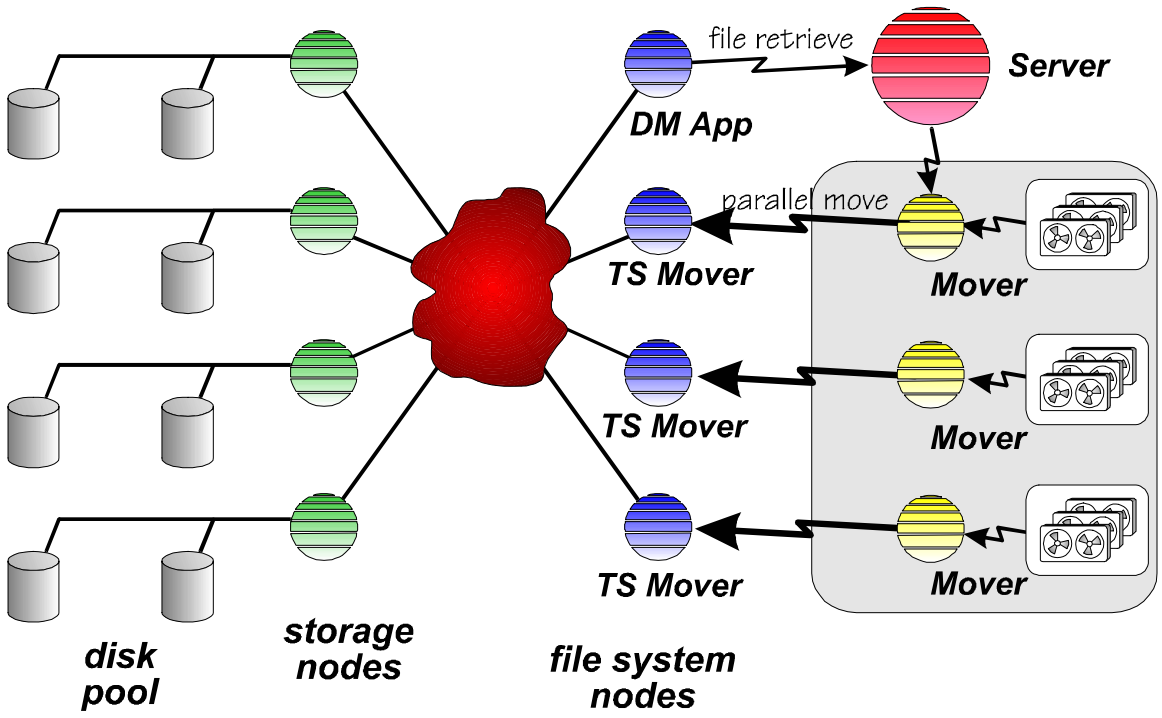
GPFS 3.1 Post-GA Work

- **Ongoing work with the SAMBA community**
  - Improve Samba/GPFS as a scalable CIFS server
    - Locking awareness between SAMBA export & GPFS use within cluster to preserve cached changes
- **SLES 10 support**
- **Healthcheck thread**
- **Explicit inode preallocation**
- **64 Filesystems**
- **Force unmount filesystems on remote clusters**
- **Fast cluster manager takeover**
- **Partial replica allocation**
- **TSM usage of pools for backup/restore**
- **GPL Licensing of all GPFS Kernel extension function**

IBM Confidential; subject to change without notice

Pre GPFS 3.1: Hierarchical Storage Management

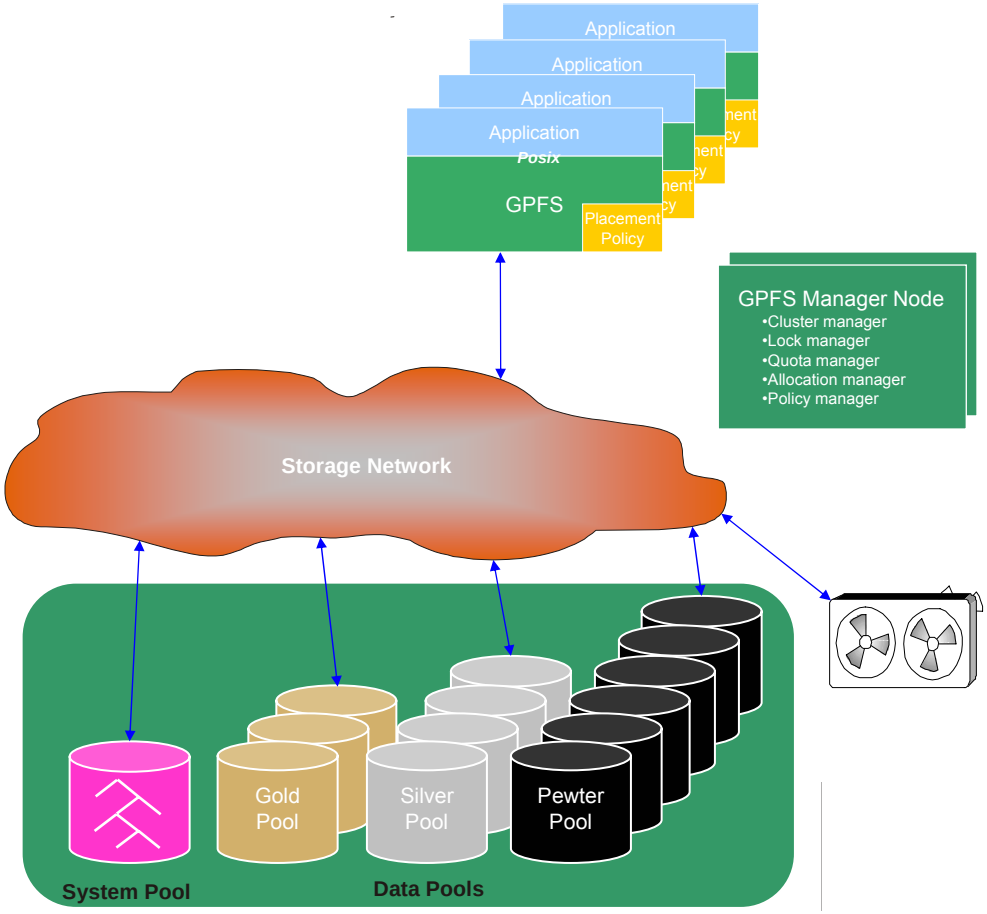
- **Archiving to tape critical to HPC**
  - File system is scratch storage
  - Tape is permanent storage
  - Automated disk-tape HSM
- **GPFS implements XOpen's XDM (aka DMAPI) API for hierarchical storage management**
  - Supported by Tivoli TSM
  - HPSS support demonstrated at SC05, GA in 2006
- **GPFS -generates storage management events**
- **DM App selects files to move to free up space**
  - File metadata for empty files remains in the file system
- **Migrated files brought back to disk when accessed**
- **"Parallel Mover" architecture provides high throughput between GPFS and parallel storage management software.**



IBM Confidential; subject to change without notice

Adding Information Lifecycle Management to GPFS

- **GPFS supports ILM via 3 new abstractions: filesets, storage pools, policy**
  - Fileset: subtree of a file system
  - Storage pool – group of LUNs
  - Policy – rules for assigning files to storage pools
- **Examples of policy rules**
  - Place new files on fast, reliable storage, move files as they age to slower storage, then to tape (in 2007)
  - Place media files on video-friendly storage (fast, smooth), other files on cheaper storage
  - Place related files together, e.g. for failure containment



IBM Confidential; subject to change without notice

Storage Pools

- **What they are:**
  - A pool is a named subset of the disks (LUNs, NSDs) in a file system
  - Each disk can be in only one pool
  - All metadata goes in default (“system”) pool
  - Each file assigned to a storage pool based upon policy rules
    - Files adjacent in the name space can be in different pools
    - … as opposed to using multiple file systems to segregate data
- **What they are good for:**
  - Tiered storage (files aged to slower/cheaper disk)
  - Dedicated storage (e.g. per user or per project or per directory subtree)
  - Failure containment
    - To limit the amount of data lost due to a failure
    - To bound the performance impact of RAID rebuild
  - Appropriate use of special-purpose storage
    - Different RAID levels
    - Enterprise-grade disk vs. consumer-grade disk
    - Multimedia-friendly storage
- **Side effects**
  - Fragmentation – file system may return E_NOSPACE even if space remains in other pools
  - Lower throughput because data is only striped across a subset of disks

IBM Confidential; subject to change without notice

Filesets

- **What they are:**
  - A *named* subtree of a GPFS file system
  - Somewhat like a distinct file system, i.e. a fileset can be unlinked without deleting it, and it can subsequently be linked using its name as a handle
- **What they are good for:**
  - Filesets can have quotas associated with them (global; not per-pool).
    - Fileset quotas are independent of user and group quotas
  - Filesets can be used to restrict the effect of policies to specific files
- **Side effects:**
  - Unlinked filesets can confuse programs that scan the file system (e.g. incremental backup programs)
  - Moving and linking between filesets is not allowed, in keeping with their being like little file systems

IBM Confidential; subject to change without notice

Policies

- **What they are:**
  - Rules to control the placement, migration, and retention of files
- **Declarative SQL-like language**
  - Rules evaluated in order
- **Rule types:**
  - **Placement** policies, evaluated at file creation, example

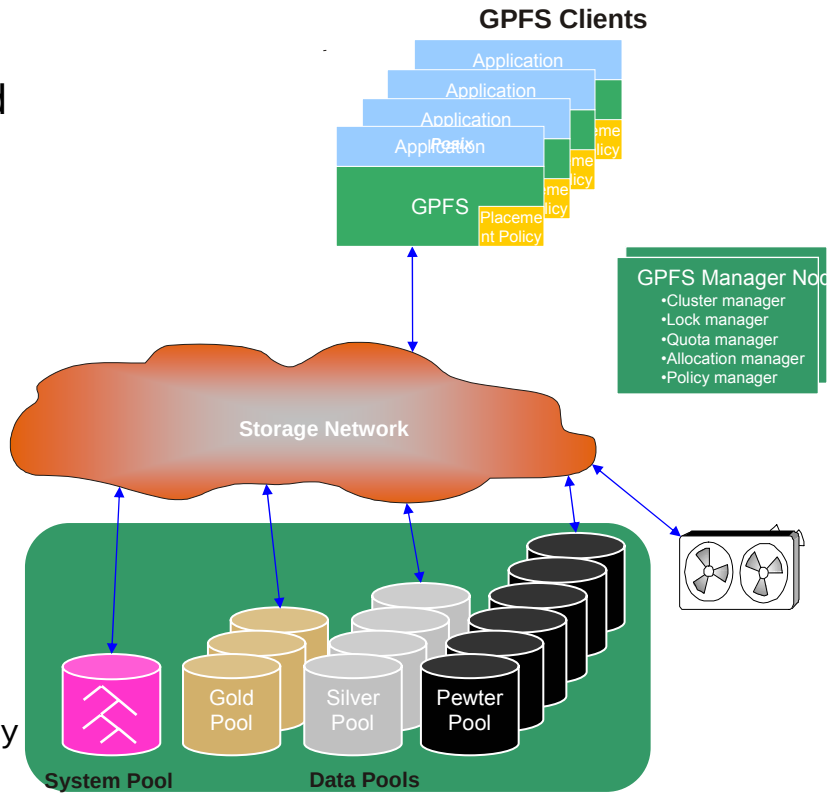
```
rule xxlfiles set pool gold for fileset
  xxlfileset
rule otherfiles set pool silver
```

  - **Migration** policies, evaluated periodically

```
rule cleangold migrate from pool gold
  threshold (90,70) to pool silver
rule cleansilver when day_of_week() = monday
  migrate from pool silver to pool pewter
  where access_age > 30 days
```

  - **Deletion** policies, evaluated periodically

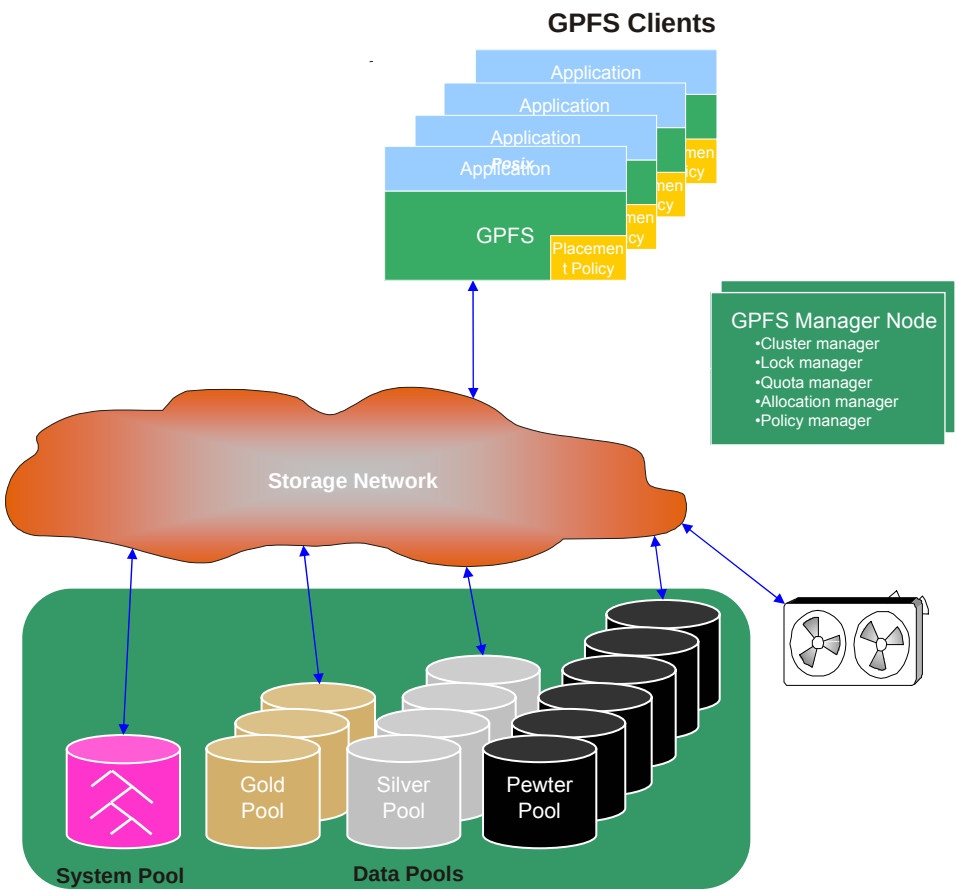
```
rule purgepewter when day_of_month() = 1
  delete from pool pewter where access_age >
  365 days
```



IBM Confidential; subject to change without notice

Placement policies

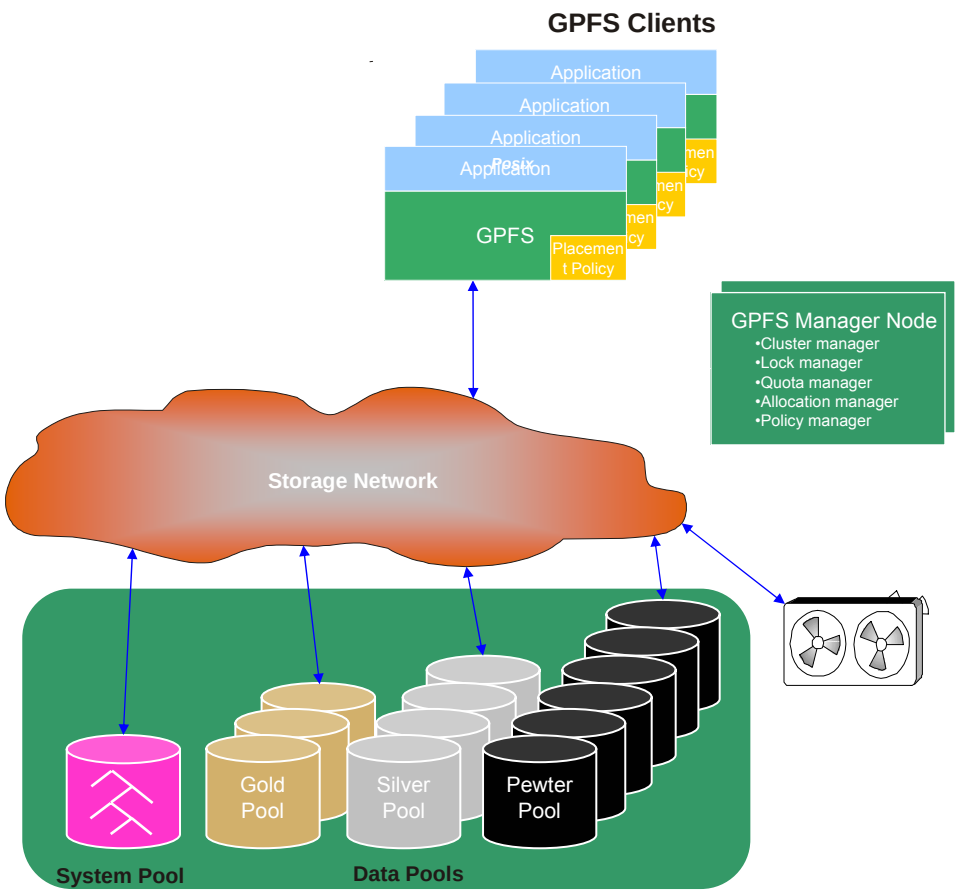
- **Admin installs policies with mmchpolicy command**
  - No per-user policies
- **Manager sends placement policies to clients at mount or after mmchpolicy**
  - Sent in internal format
  - Necessary for multicluster
- **Client autonomously evaluates placement policies at file create**



IBM Confidential; subject to change without notice

Migration and Deletion Policies

- **Evaluated by executing new mmapplypolicy command**
  - Can be invoked periodically, e.g. from cron job
  - … or explicitly by administrator
- **Scans all files, applying each rule in turn to each file**
  - Files to be migrated are marked
  - Files to be deleted are deleted during scan
- **Actual migration is done in parallel at end of scan**
  - new mmrestripefs –p option

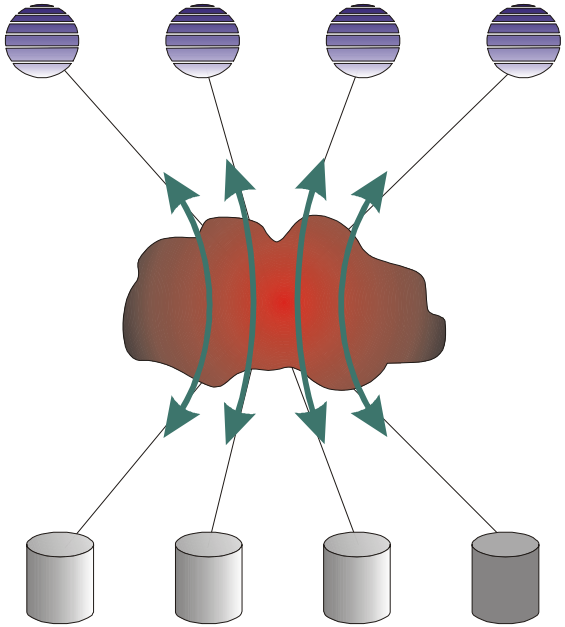


IBM Confidential; subject to change without notice

Summary

GPFS

- **… gives a cluster of machines the image of a single shared file system**
- **… has performance and capacity that scales to the largest problems**
- **… is fully robust and fault-tolerant to support line-of-business applications**
- **… for all its capacity and performance, is simple to manage**
- **… rapidly broadening from its HPC roots into a data intensive world**



IBM Confidential; subject to change without notice

GPFS Documentation

- **Samples files:**
  - /usr/lpp/mmfs/samples/
- **GPFS Documentation**
  - http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfsbooks.html
- **GPFS Frequently Asked Questions (FAQ)**
  - http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfs_faqs.html
- **GPFS General Documentation (including older white paper)**
  - http://www-03.ibm.com/servers/eserver/clusters/software/gpfs.html
- **UID Mapping for GPFS in a Multi-cluster Environment**
  - http://www-03.ibm.com/servers/eserver/clusters/whitepapers/uid_gpfs.html
- **Disaster Recovery with GPFS**
  - http://www-03.ibm.com/servers/eserver/pseries/software/whitepapers/gpfs_recovery.html
- **Clusters Library**
  - www.ibm.com/servers/eserver/clusters/library/wp_aix_lit.html

IBM Confidential; subject to change without notice

Questions?

Thank you!

IBM Confidential; subject to
change without notice