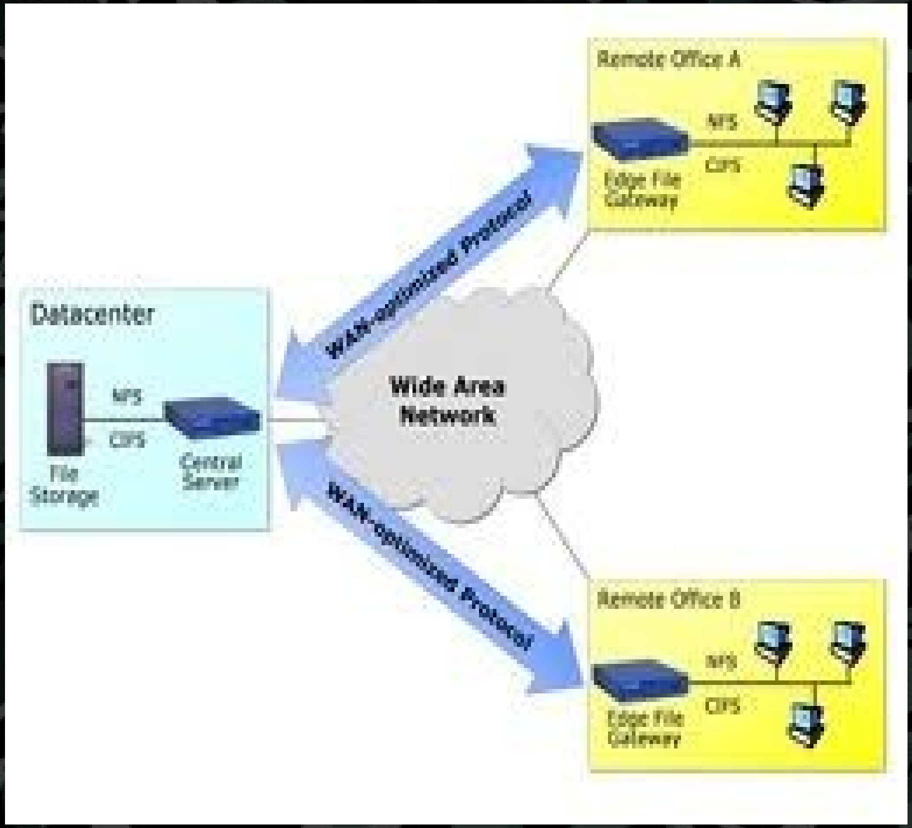
HPC Challenge in Distributed and Parallel File systems



- Collaboration Summit 2011
- Christoph Lameter, Linux Foundation

The Distributed Scaling Problem

- Data has to be available everywhere
- Large volume of data has to be accessed very fast
- Failure of any component should not lead to loss of data

- Distributed files problem
- Throughput problem
- Resiliency
- 24hr availability
- I want my data always anywhere with high speed and no data loss.

A whining session

- Users feel these requirements are reasonable and it should be simple to create something that does this.
- Reality is that none of that works and striving for these goals is a constant headache for IT departments.
- Heard this from multiple places
- No real projects to address all of these issues

RAID handling or redundancy

- Differs between systems
  - RAID on volumes
  - RAID over nodes
  - RAID on a per file basis
- Hardware and Software Raid

Hotspot avoidance

- 1000s of machine accessing a single file.
- Some FS can do replication for this (Ceph f.e.)
- In some environments this is handled at the block layer (HPs 3PAR)
- Caching the object for that purpose
- But it could be a pretty large file that is read by all.
- Hotspots across a WAN are a particular issue.

Throughput

- So far been able to bring everything vendors threw at us to its knees
- 10GB-100GB/sec mininum.
- Special high speed interconnect
- Difficulty to get hardware vendors to believe us.
- FS problems
- Instability issues with OS, drivers and hardware.

The ideal world

- File will automatically be migrated globally to wherever a file is going to be used
- Multiple redundant file servers that can fail without impacting reachable.
- Global Filesystem: One path reaches the file that I want from everywhere I could be.
- Binaries and script can run everywhere without change.

Islands of Solutions

- Local multi node file systems
  - Lustre/Gluster
  - Ceph
  - GPFS/IRIX/CXFS
- Global Filesystems
  - AFS/CodaFS
  - ExtreemFS
  - OpenEFS
  - Dcache/GFS

GFS - GoogleFS

- Single master metadata server
- Chunk servers as storage nodes
- Append only write semantics
- Not POSIX compliant
- Customized to Googles need.
- Seems to be designed with some WAN access in mind

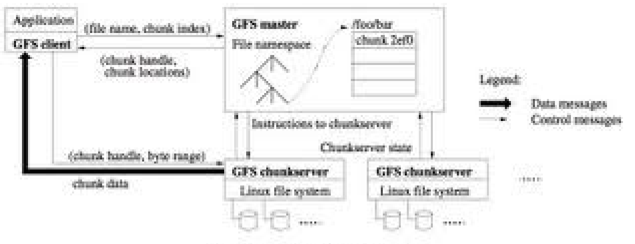


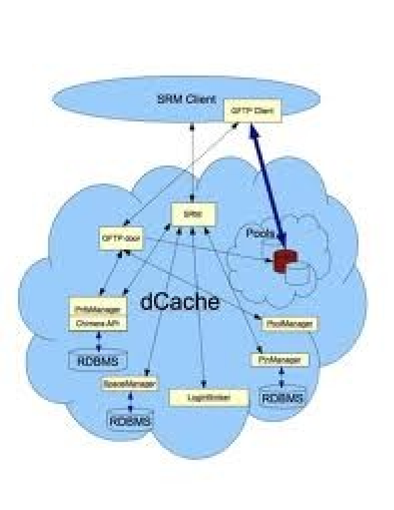
Figure 1: GFS Architecture

XtreemFS

- Distributed filesystem
- But Grid focus
- Client caches only metadata
- Early development
- Major features like read-write files with POSIX compliance, snapshots etc may take a long time.

dCache – Lab

- Written for huge data streams
- POSIX compliant
- Http://www.dcache.org/manuals/dcache-whitepap
- Tertiary Storage support
- Replication via WAN

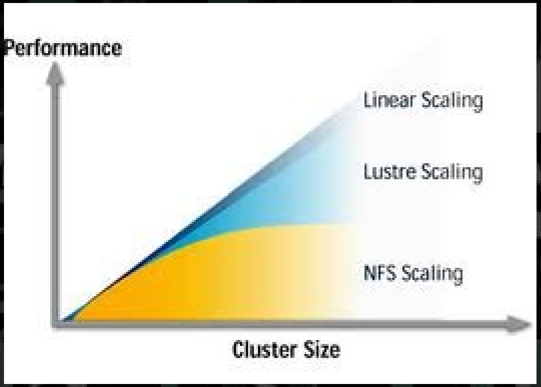


OpenEFS

- Focus on versioning
- Perl based scripts to maintain archives across a WAN
- Common namespace
- Conflict with packaging system
- Solution for application build consistency.

Lustre

- Good for performance (fastest...)
- Less so for reliability
- No operation across a WAN
- Complicated kernel patching (out of tree) especially if used with Infiniband support

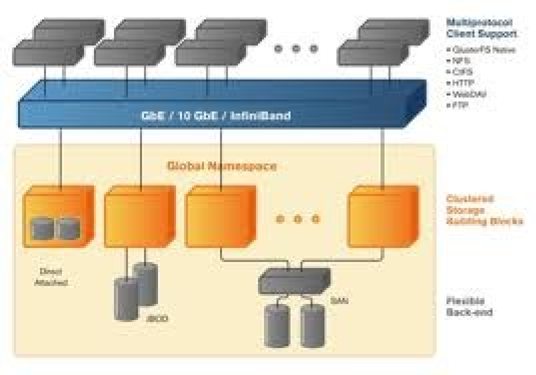


Ceph

- Resiliency
- Manages redundancy and distribution over multiple nodes itself.
- Migrates files to where they are used.
- Authors do not want to deal with WAN issues
- Depends on btrfs and btrs is not yet production ready.
- Good product at some future date.

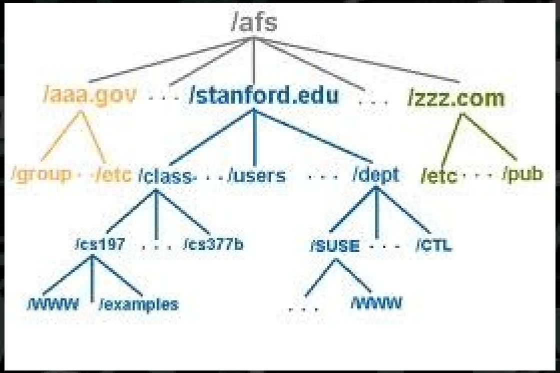
Gluster

- Ingenious solution that works based on filename translators.
- No WAN support
- Very fast in recent versions
- Easiest to deploy since there is less dependency on low level filesystems.
- Aims to be a small layer at the top.



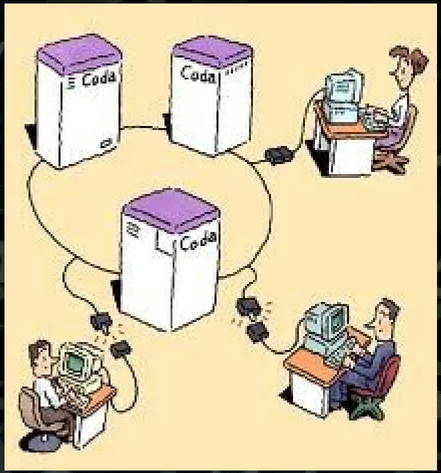
AFS

- Very established distributed filesystem solution from the 80s.
- Easy replication of read-only data
- Only a single writeable copy.
- Trouble with updating files
- Suitable for large scale
- deployments
- Not a “parallel” filesystem.



CodaFS

- Solves the write issues of AFS.
- Disconnected operations
- File is moved to client that accesses it.
- Resiliency?
- Not a parallel filesystem



Proprietary Solutions

- Need to load large binary blobs into the kernel
- Licensing fees per node
- Trouble with building your own kernel
- In practice this leads to deployment only for special systems.
- Reexport via NFS, CIFS is common
- None of them does really support distributing files across WAN.
- Proprietary solutions are present because there is no compelling open source solution.

GPFS (IBM)

- LAN only
- Useful for general use: Enterprise class reliability but still good performance. POSIX semantics.
- Versatile configuration
- Preconfigured systems and services ("Scale-out File Services")

IBRIX (HP)

- Filesystem
- Lately becomes bricked (appliance) in form of the X9000

3PAR (HP)

- Superior hotspot avoidance
- Compressions (avoid duplication of blocks that have the same content)
- Self maintaining
- Its more of a block device though.

CXFS (SGI)

- HPC orientation
- Focus on high performance over against enterprise class reliability
-

Where to go from here

- All solutions are a bit complicated and are not full solutions
- Complexity of such an endeavor
- Integration of host based FS, inter node FS and WAN manager.
- Can we coordinate multiple projects to tackle this?