SciDAC Petascale Data Storage Institute
Advanced Scientific Computing Advisory Committee Meeting
October 29 2008, Gaithersburg MD

Garth Gibson
Carnegie Mellon University and Panasas Inc.
SciDAC Petascale Data Storage Institute (PDSI)
www.pdsi-scidac.org

w/ LANL (Gary Grider), LBNL (William Kramer), SNL (Lee Ward),
ORNL (Phil Roth), PNNL (Evan Felix),
UCSC (Darrell Long), U.Mich (Peter Honeyman)
Charting Storage Path thru Peta- to Exa-scale

- Top500.org scaling 100% per year; storage required to keep up
  - This is hard for disks: MB/sec +20% per year, accesses/sec +5% per year
  - Increases number of disks much faster than processor chips or nodes
SciDAC Petascale Data Storage Institute

- High Performance Storage Expertise & Experience
  - Carnegie Mellon University, Garth Gibson, lead PI
  - U. of California, Santa Cruz, Darrell Long
  - U. of Michigan, Ann Arbor, Peter Honeyman
  - Lawrence Berkeley National Lab, William Kramer
  - Oak Ridge National Lab, Phil Roth
  - Pacific Northwest National Lab, Evan Felix
  - Los Alamos National Lab, Gary Grider
  - Sandia National Lab, Lee Ward
SciDAC Petascale Data Storage Institute

• Efforts divided into three primary thrusts
• Outreach and leadership
  • Community building: ie. PDSW @ SC08, FAST, FSIO
  • APIs & standards: ie., Parallel NFS, POSIX Extensions
  • SciDAC collaborations: applications, centers, institutes
• Data collection and dissemination
  • Failure data collection, analysis: ie., cfdr.usenix.org
  • Performance trace collection & benchmark publication
• Mechanism innovation
  • Scalable metadata, archives, wide area storage, etc
  • IT automation applied to HEC systems & problems
Outreach: Sponsored Workshops

- SC08: PDSW, Mon Nov 17, 8-5

Supercomputing '07
Petascale Data Storage Workshop
Session Chair: Garth Gibson, CMU
Sunday, November 11, 2007
Reno, Nevada

WORKSHOP ABSTRACT
Petascale computing infrastructures make petascale demands on information and management. The last decade has shown that parallel file systems are not enough. This poses a critical challenge when we look at the data storage problems and emerging solutions found in petascale scale. New community collaboration is crucial, problem identification, workload evaluation.

Petascale Data Storage Workshop Introduction
Garth Gibson

SESSION I: Scalable Systems

SESSION II: Scalable Services
Jonathan Koren (presenter), Y. Zhang, Univ. of California, Santa Cruz

POSTER SESSION 1 - see info below

November

February

FAST '08
Petascale Data Storage BOF Session at FAST '08

Wednesday, February 27, 2008
The Computer Failure Data Repository (CFDR) - Bianca Schneider, University of Toronto
The File System Statistics - Shobhit Dayal, CMU, Garth Gibson, CMU, Merc Unzangut, Panasas
PNNL - Petascale Data Storage Institute Data release Update - Evan Finkel, PNNL
NERSC Reliability Data - Bill Kramer, Jason How, Akkoyunlu, NERSC
LANL/SuDAC Petascale Data Storage Institute Operational Data Release - James Unzangut, Gary Erickson, John Bent, HB Chen, Meghan Guilford, Nelis Thomas, Los Alamos National Lab
Gem: An Open-Source PetaScale File System - Ethan Miller, Storage Systems Research Center, UC Santa Cruz

Special Presentation on HPC User Requirements:
IG Requirements for PDC Applications: A User Perspective
John Shalt, National Energy Research Scientific Computing Center (NERSC), LBNL

POSTER PRESENTATION ABOUT THE FAST '08 POSTER SESSION
PNNL Data Releases and Repositories

Visit www.pdsi-scidac.org/SC07/ for more information.
Standards: Multi-vendor, Scalable Parallel NFS

- Persistent investment in scalability
  - Share costs with commercial R&D
- Next generation NFS is parallel (pNFS)
  - Responds to growing role of clusters
  - Open source & competitive offerings
  - NetApp, Sun, IBM, EMC, Panasas, BlueArc, DESY/dCache

From: Spencer Shepler <Spencer.Shepler@Sun.COM>
Date: August 1, 2008 4:34:46 PM GMT-04:00

2. IETF status

All of the current working group internet drafts are moving forward for publication. This means that they have submitted to the area director and will start their way through the process (IETF last call and IESG review).
Dissemination: Fault Data

- Los Alamos root cause logs
- 22 clusters & 5,000 nodes
- covers 9 years & continues
- cfdr.usenix.org publication + PNNL, NERSC, Sandia, PSC, …

![Bar chart showing failures per year per proc]

<table>
<thead>
<tr>
<th>(I) High-level system information</th>
<th>(II) Information per node category</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW</td>
<td>ID</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Overview of Los Alamos SMP-based, and system
Projections: More Failures

- Con’t top500.org 2X annually
  - 1 PF Roadrunner, May 2008
- Cycle time flat, but more of them
  - Moore’s law: 2X cores/chip in 18 mos
- # sockets, 1/MTTI = failure rate up 25%-50% per year
  - Optimistic 0.1 failures per year per socket (vs. historic 0.25)
Fault Tolerance Challenges

- Periodic \( (p) \) pause to checkpoint \( (t) \)
  - Major need for storage bandwidth
- Balanced systems
  - Storage speed tracks FLOPS, memory
    so checkpoint capture \( (t) \) is constant
  - \( 1 - \text{AppUtilization} = \frac{t}{p} + \frac{p}{2\times\text{MTTI}} \)
    \[ p^2 = 2t\times\text{MTTI} \]

- but dropping MTTI
  kills app utilization!

---

```
Everything Must Scale with Compute
```

- Computing Speed
  - TFLOP/s
- Parallel I/O
  - TeraBytes
  - GigaBytes/sec
- Network Speed
  - Gigabits/sec
- Archival Storage
  - TeraBytes
  - PetaBytes
- Disk
  - GigaBytes/sec
  - PetaBytes
- Metadata
  - Inserts/sec

---

```
Carnegie Mellon
```

www.pdsi-scidac.org
Bolster HEC Fault Tolerance

- More storage bandwidth?
  - disk speed 1.2X/yr
    - # disks +67%/y
      just for balance!
  - to also counter MTTI
    - # disks +130%/yr!
  - Little appetite for the cost

- Compress checkpoints!
  - plenty of cycles available
  - smaller fraction of memory
each year (application specific)
    - 25-50% smaller per year
Alternative Approaches

- Dedicated checkpoint device (ie., PSC Zest)
  - Stage checkpoint through fast memory
  - Cost of dedicated memory large fraction of total
  - Cheaper memory (flash?) now bandwidth limited

- Classic enterprise process pairs duplication
  - Flat 50% efficiency cost, plus message duplication
## Storage Suffers Failures Too

<table>
<thead>
<tr>
<th>System</th>
<th>Type of Drive</th>
<th>Count</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pittsburgh Supercomputing Center</td>
<td>HPC1 18GB 10K RPM SCSI 36GB 10K RPM SCSI</td>
<td>3,400</td>
<td>5 yrs</td>
</tr>
<tr>
<td></td>
<td>HPC2 36GB 10K RPM SCSI</td>
<td>520</td>
<td>2.5 yrs</td>
</tr>
<tr>
<td>Los Alamos National Laboratory</td>
<td>HPC3 15K RPM SCSI 15K RPM SCSI 7.2K RPM SATA</td>
<td>14,208</td>
<td>1 yr</td>
</tr>
<tr>
<td>Supercomputing X</td>
<td>HPC4 250GB SATA 500GB SATA 400GB SATA</td>
<td>13,634</td>
<td>3 yrs</td>
</tr>
<tr>
<td>Various HPCs</td>
<td>COM1 10K RPM SCSI</td>
<td>26,734</td>
<td>1 month</td>
</tr>
<tr>
<td>Internet services Y</td>
<td>COM2 15K RPM SCSI</td>
<td>39,039</td>
<td>1.5 yrs</td>
</tr>
<tr>
<td></td>
<td>COM3 10K RPM FC-AL 10K RPM FC-AL 10K RPM FC-AL</td>
<td>3,700</td>
<td>1 yr</td>
</tr>
</tbody>
</table>
Storage Failure Recovery is On-the-fly

- Scalable performance = more disks
- But disks are getting bigger
- Recovery per failure increasing
- Hours to days on disk arrays
- Consider # concurrent disk recoveries
  - e.g. 10,000 disks
  - 3% per year replacement rate
  - 1+ day recovery each
  - Constant state of recovering?
- Maybe soon 100s of concurrent recoveries (at all times!)
- Design normal case for many failures (huge challenge!)
Parallel Scalable Repair

- Defer the problem by making failed disk repair a parallel app
- File replication and, more recently, object RAID can scale repair
  - “decluster” redundancy groups over all disks (mirror or RAID)
  - use all disks for every repair, faster is less vulnerable
- Object (chunk of a file) storage architecture dominating at scale
  PanFS, Lustre, PVFS, … GFS, HDFS, … Centera, …
PDSI Collaborations

• Primary partner: Scientific Data Management (SDM)
  – PVFS, metadata, checkpoint, failure diagnosis
• Storage IO & Performance Engineering (PERI)
  – Ocean (POP), Combustion (S3D), P. Roth
• Storage trouble shooting with trace tools
  – Climate Change (CCSM), L. Ward
• Vendor partnerships & centers of excellence
  – pNFS, IBM GPFS, Sun Lustre, Panasas, EMC
• Leadership computing facilities partnerships
  – Roadrunner, Redstorm, Jaguar, Franklin, ...
• Base program cooperation: FASTOS IO Forwarding
Its All About Data, Scale & Failure

• Continual gathering of data on data storage
  – Failures, distributions, traces, workloads
• Nurturing of file systems to HPC scale, requirements
  – pNFS standards, benchmarks, testing clusters, academic codes
• Checkpoint specializations
  – App-compressed state, special devices & representations
• Failure as the normal case?
  – Risking 100s of concurrent disk rebuilds (need faster rebuild)
  – Quality of service (performance) during rebuild in design
• Correctness at increasing scale?
  – Testing using virtual machines to simulate larger machines
  – Formal verification of correctness (performance?) at scale
• HPC vs Cloud Storage Architecture
  – Common software? Share costs with new technology wave
SciDAC Petascale Data Storage Institute
Advanced Scientific Computing Advisory Committee Meeting,
October 29 2008, Gaithersburg MD

Garth Gibson
Carnegie Mellon University and Panasas Inc.

SciDAC Petascale Data Storage Institute (PDSI)

www.pdsi-scidac.org

w/ LANL (Gary Grider), LBNL (William Kramer), SNL (Lee Ward),
ORNL (Phil Roth), PNNL (Evan Felix),
UCSC (Darrell Long), U.Mich (Peter Honeyman)
Tools: Understanding IO in Apps

NEWEST TRACE DATA, REDSTORM, SANDIA NAT’L LAB
- A physics simulation problem for a common Sandia application, Alegra
- Runs were performed alongside regular user runs
- Each run generated 4 restart dumps, and ran for 20 simulation cycles
- Both single core per node, and 2 core (virtual node mode) per node
  - Repeated with and without tracing enabled
- The single core per node jobs ran at a client size of 2744 processes
  - Non-tracing elapsed run time 10:42 minutes
  - Tracing elapsed run time 11:07 minutes
- The 2 core per node jobs ran at 2916 nodes, 5832 processes.
  - Non-tracing elapsed run time 15:52 minutes
  - Tracing elapsed run time 16:37 minutes
- Raw trace file sizes 30K-50K per MPI rank, except rank zero (600KB-700KB)
- Rank 0 I/O to terminal records progress in the job.

sourceforge.net/projects/libsysio
Dissemination: Parallel Workloads

**NEWEST TRACE DATA, REDSTORM, SANDIA NAT’L LAB**
- A physics simulation problem for a common Sandia application, Alegra
  - Runs were performed alongside regular user runs
  - Each run generated 4 restart dumps, and ran for 20 simulation cycles
- Both single core per node, and 2 core (virtual node mode) per node
  - Repeated with and without tracing enabled
- The single core per node jobs ran at a client size of 2744 processes
  - Non-tracing elapsed run time 10:42 minutes
  - Tracing elapsed run time 11:07 minutes
- The 2 core per node jobs ran at 2916 nodes, 5832 processes.
  - Non-tracing elapsed run time 15:52 minutes
  - Tracing elapsed run time 16:37 minutes
- Raw trace file sizes 30K-50K per MPI rank, except rank zero (600KB-700KB)
  - Rank 0 I/O to terminal records progress in the job.

**I/O calls, 2744 Processes**

**I/O Transfers, 2744 Processes**

**I/O Transfers, 5832 Processes**

sourceforge.net/projects/libsysio
Dissemination: Parallel Workloads

**MPI-IO Test**

Although there are a host of existing file system and I/O test programs, most are not designed with parallel I/O in mind and are not useful at the clusters at Los Alamos National Lab (LANL). LANL’s MPI-IO Test was built with parallel I/O and scale in mind. The MPI-IO test is built on top of MPI-1 and is used to gather timing information for reading from and writing to files using a variety of I/O profiles; N processes writing to N files, N processes sending to one file, N processes sending data to M processes writing to M files, etc. These diagrams illustrate various I/O access patterns. A data aggregation capability is available and can pass down MPI-IO, ROMIO and file system specific hints. The MPI-IO Test can be used for performance benchmarking and, in some cases, to diagnose problems with file systems or I/O networks.

The MPI-IO Test is open sourced under LA-CC-05-013.

<table>
<thead>
<tr>
<th>Release</th>
<th>Date</th>
<th>Source</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000.21</td>
<td>8 July 2008</td>
<td>mpi_io_test_21.tgz</td>
<td>README</td>
</tr>
<tr>
<td>1.000.20</td>
<td>13 November 2007</td>
<td>mpi_io_test_20.tgz</td>
<td>README</td>
</tr>
<tr>
<td>1.000.09</td>
<td>15 December 2006</td>
<td>mpi_io_test_09.tgz</td>
<td>README</td>
</tr>
<tr>
<td>1.000.08</td>
<td>2 March 2006</td>
<td>mpi_io_test_08.tgz</td>
<td>README</td>
</tr>
</tbody>
</table>
Dissemination: Parallel Workloads

**MADBech: Microwave Anisotropy Dataset Computational Analysis Package Benchmark**
The benchmark code MADBech is a "stripped-down" version of MADCAP, a Microwave Anisotropy Dataset Computational Analysis Package...more>>>

**I/P benchmarks**: Medium, Large and X-large datasets.

**MILC: MIMD Lattice Computation**
The benchmark code MILC represents part of a set of codes written by the MIMD Lattice Computation (MILC) collaborator used to study quantum chromodynamics (QCD), the theory of the strong interactions of subatomic physics...more>>>

**I/P benchmarks**: Medium and Large datasets.

**PMEMD: Particle Mesh Ewald Molecular Dynamics**
The benchmark code PMEMD (Particle Mesh Ewald Molecular Dynamics (MD), NMR Refinement and minimizations...more>>>

**I/P benchmarks**: Medium and Large datasets

**IO Benchmarks with IPM**
The new version of IPM integrates the standard POSIX I/O call runs are made with this new feature on Jacquard (courtesy of GARFIELD)

**MADBech**:
- 256 tasks, POSIX one file per task [plots] [stats]
- 64 tasks, POSIX one file per task [plots] [stats]
- 16 tasks, POSIX shared file [plots] [stats]

**Chombo**:
- 256 tasks, 2 components [plots] [stats]
- 32 tasks, 2 components [plots] [stats]
- 32 tasks, 10 components [plots] [stats]

**AMRScalingXfer**: 128 tasks, small run [plots] [stats]

*Note: This is development software, and the runs/plots are not profiling in IPM.

**I/O Benchmark and Characterization Links:**

**I/O Performance for HPC Platform using IOR**[PDF][ppt]
This study analyzes the I/O practices and requirements of current HPC applications and uses them as criteria to select a subset of microbenchmarks that reflect workload requirements.

**FLASH I/O Benchmark**[PDF]
This code from 'The Center for Astrophysical Thermonuclear Flashes' can test either HDFS, Parallel NetCDF, or a direct Fortran write. The I/O benchmarks are compared for Seaborg and Bassi systems.

**Performance Effect of Multi-core on Scientific Applications** (PDF)[paper slides]
Performance measurements of several complete scientific applications on single and dual core Cray XT3 and XT4 systems.

**MADBech - IPM of a Cosmology Application on Leading HPC Platforms**[PDF]
Presents MADBech, a lightweight version of MADCAP CMB power spectrum estimation code, and uses the Integrated Performance Monitoring (IPM) package to extract MPI message-passing overheads.

**MADBech2**[PDF]
Presents I/O analyses of modern parallel filesystems and examines a broad range of system architectures and configurations. It also describes use of Luster striping to improve concurrent file access performance.

**Effective I/O Bandwidth Benchmark**[PDF]
This paper describes the design and implementation of a parallel I/O benchmark useful for comparing filesystem performance on a variety of architectures, including, but not limited to cluster systems.

**Efficient Parallel I/O on the Cray XT3/XT4**[PDF]
Provides an overview of I/O methods for three different applications.

**Trace Data**
Here are files containing trace data for some of the applications. These traces are generated by invoking the "strace" utility on every task and piping the data for each task to a separate file. Process ID is used to create unique file names. All applications were run on Jacquard. The files are compressed tar files of the trace data.

- **PMEMD**: 16 tasks small dataset run
- **MADBech**: 64 tasks medium dataset run
- **MILC**: 16 tasks medium dataset run
Dissemination: Statistics
Mechanisms for Scalable Metadata (1)

Spyglass design

- Partition file system hierarchy by subtree
  - Each subtree is an independent subindex
- Summarize contents of each subindex
  - Quickly rule out entire subindexes that can’t satisfy the query
- Log incremental changes
  - Rebuild index when there are “enough” changes
- Integrity is much easier
  - Rebuild subindex, not entire index
Mechanisms for Scalable Metadata (2)

- Billion+ files in a directory
- Eliminate serialization
  - All servers grow directory independently, in parallel, without any co-ordinator

Local representation of huge directory in Giga+

- No synchronization or consistency bottlenecks
  - Servers only keep local “view”, no shared state

![Diagram showing local representation of huge directory](image)

Scale and performance of Giga+ using UCAR Metarates benchmark.

Garth Gibson, 10/29/2008