DOE HPC Best Practices Workshop

File Systems and Archives

U.S. DEPARTMENT OF ENERGY

Office of Science

SEPTEMBER 26-27, 2011 • SAN FRANCISCO, CA

The Fifth Workshop on HPC Best Practices: File Systems and Archives

Held September 26–27, 2011, San Francisco

Workshop Report

December 2011

Compiled by Jason Hick, John Hules, and Andrew Uselton
Lawrence Berkeley National Laboratory

Workshop Steering Committee
John Bent (LANL); Jeff Broughton (LBNL/NERSC); Shane Canon (LBNL/NERSC); Susan Coghlan (ANL); David Cowley (PNNL); Mark Gary (LLNL); Gary Grider (LANL); Kevin Harms (ANL/ALCF); Paul Henning, DOE Co-Host (NNSA); Jason Hick, Workshop Chair (LBNL/NERSC); Ruth Klundt (SNL); Steve Monk (SNL); John Noe (SNL); Lucy Nowell (ASCR); Sarp Oral (ORNL); James Rogers (ORNL); Yukiko Sekine, DOE Co-Host (ASCR); Jerry Shoopman (LLNL); Aaron Torres (LANL); Andrew Uselton, Assistant Chair (LBNL/NERSC); Lee Ward (SNL); Dick Watson (LLNL).

Workshop Group Chairs
Shane Canon (LBNL); Susan Coghlan (ANL); David Cowley (PNNL); Mark Gary (LLNL); John Noe (SNL); Sarp Oral (ORNL); Jim Rogers (ORNL); Jerry Shoopman (LLNL).

Ernest Orlando Lawrence Berkeley National Laboratory
1 Cyclotron Road, Berkeley, CA 94720-8148

This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

Contents

Executive Summary

In 2006, the United States Department of Energy (DOE) National Nuclear Security Administration (NNSA) and Office of Science (SC) identified the need for large supercomputer centers to learn from each other on what worked well and what didn't in the areas of acquisition, installation, integration, testing and operation. Previous High Performance Computing (HPC) Best Practice Workshops focused on System Integration in 2007 (http://outreach.scidac.gov/pibp/), Risk Management in 2008 (http://rmtap.llnl.gov/), Software Lifecycles in 2009 (http://outreach.scidac.gov/swbp/), and Power Management in 2010 (http://outreach.scidac.gov/pmbp/).

The workshop on HPC Best Practices on File Systems and Archives was the fifth in the series. The workshop gathered technical and management experts for operations of HPC file systems and archives. Attendees identified and discussed best practices in use at their facilities, and documented findings for the DOE and HPC community in this report. The home page for this workshop is http://outreach.scidac.gov/fsbp/.

Prior to the workshop, board members identified four critical functionally components to the operation of storage systems: administration, business, reliability, and usability of storage. These components of storage became the breakout session topics. During registration, attendees designated their primary interests for breakout sessions and submitted a position paper to aid in having discussions around proposals or ideas in the position papers provided. Participants submitted 27 position papers, included in Appendix A.

This year the workshop had 64 attendees from 29 different sites from around the world. Participants identified 16 best practices in use:

1. For critical data, multiple copies should be made on multiple species of hardware in geographically separated locations.
2. Build a multidisciplinary data center team.
3. Having dedicated maintenance personnel, vendor or internal staff, is important to increasing system availability.
4. Establish and maintain hot-spare, testbed, and pre-production environments.
5. Establish systematic and ongoing procedures to measure and monitor system behavior and performance.
6. Establish authoritative sources of information.
7. Manage users' consumption of space.
8. Storage systems should function independently from compute systems.
9. Include middleware benchmarks and performance criteria in system procurement contracts.
10. Deploy and support tools to enable users to more easily achieve the full capabilities of file systems and archives.
11. Training, onsite or online, and meeting with the users directly improves the utilization and performance of the storage system.
12. Actively manage relationships with the storage industry and storage community.
13. Monitor and exploit emerging storage technologies.
14. Continue to re-evaluate the real characteristics of your I/O workload.
15. Use quantitative data to demonstrate the importance of storage to achieving scientific results.
16. Retain original logs generated by systems and software.

Many participants do not implement all the best practices, which means they are bringing several new ideas for improvement back to their sites. In addition, attendees identified 15 gaps that require further attention. Gaps likely addressed by evolutionary solutions include:

1. Incomplete attention to end-to-end data integrity.
2. Storage system software is not resilient enough.
3. Redundant Array of Independent Tape (RAIT) does not exist today.
4. Provide monitoring and diagnostic tools that allow users to understand file system configuration and performance characteristics and troubleshoot poor I/O performance.
5. Communication between storage systems users and storage system experts needs improvement.
6. Need tools and mechanisms to capture provenance and provide life-cycle management for data.
7. Ensure storage systems are prepared to support data-intensive computing and new workflows.
8. Measure, and track trends in storage system usage to the same degree we measure and track flops and cycles on computational systems.
9. Tools to provide scalable performance in interacting with files in the storage system.

Gaps requiring revolutionary approaches include:

1. Need quality of service for users in a shared storage environment.
2. Need standard metadata for users to specify data importance and retention.
3. The diagnostic information currently available in today's storage systems is woefully inadequate.
4. Metadata performance in storage systems already limits usability and won't meet the needs of exascale.
5. POSIX isn't expressive enough for the breadth of application I/O patterns.
6. There exists a storage technology gap for exascale.

Introduction

The U.S. Department of Energy (DOE) has identified the design, implementation, and usability of file systems and archives as key issues for current and future high performance computing (HPC) systems. This workshop was organized to address current best practices for the procurement, operation, and usability of file systems and archives. Furthermore, the workshop addressed whether system challenges can be met by evolving current practices.

Workshop Goals

The workshop organizers presented the following goals:

- Foster a shared understanding of file system issues in the context of HPC centers.
- Identify top challenges and open issues.
- Share best practices and lessons learned.
- Establish communication paths for managerial and technical staff at multiple sites to continue discussion on these topics.
- Discuss roles and benefits of HPC stakeholders.
- Present findings to DOE and other stakeholders.

Workshop Format

The organizers requested a short position paper from each attendee to identify best practices in the area of file systems and archives. Each position paper addressed a topic or topics related to best practices for the session they were attending. The session organizers identified the questions below as key topics for each session and suggested that each paper deal with one or more of them. Participants were asked to frame their discussion to identify what they think are best practices in use at their site related to the session topics, and to frame questions within the discussion to help elicit best practices from the other participants. Position papers are collected in Appendix A.

The workshop began with presentations by representatives of the DOE sponsors: Thuc Hoang of the National Nuclear Security Administration (NNSA) and Yukiko Sekine of the Office of Science (SC). The workshop focused primarily on breakout sessions in which participants presented and evaluated selected topics from their position papers. Each day ended with a member of each breakout session presenting the day's results to the entire group. The agenda is presented in Appendix B.

Workshop Breakout Topics

In their position papers and breakout discussions, participants were asked to consider the topics listed below.

The Business of Storage Systems

For each layer of the storage hierarchy — file systems, archives, others — address these topics (and add to them):

1. HPC facilities across DOE deploy and operate systems at unprecedented scale requiring advanced file system technologies. Achieving the requisite level of performance and scalability from file systems remains a significant challenge.

    A. What are your practices used to plan for future system deployments and system evolution over time?
    B. How do you establish requirements such as bandwidth, capacity, metadata operations/sec, mean time to interrupt (MTTI), etc. for these systems?

2. It is not uncommon for archival storage deployments to have life spans approaching multiple decades. Growth of archival data at a number of HPC sites is exponential.
    A. How do you effectively plan for exponential growth rates and archives that will need to serve multiple generations of machines throughout their life within a fixed budget profile?
    B. Are exponential growth rates sustainable? How do you mitigate if not?

3. There are relatively few alternatives in parallel file system and archival storage system software that meet the requirements of major HPC facilities. Development of this software varies from proprietary closed source to collaborative open source solutions. Each model has benefits and drawbacks in terms of total cost of ownership, ability to evolve the software to meet specific requirements, and long-term viability (risk) of the system.
    A. What model do you leverage for your file system or archival system software needs?
    B. What benefits/drawbacks do you see with these models?

4. Storage system and tape archive technologies vary from high-end custom hardware developed specifically for the HPC environment to commodity storage platforms with extremely broad market saturation.
    A. Where do you leverage custom versus commodity storage hardware within your operational environment?
    B. Do you see opportunities to incorporate more commodity storage technologies within your environment in the future?
    C. What are the barriers to adopting commodity storage technologies and how can they be overcome in the future?

The Administration of Storage Systems

For each layer of the storage hierarchy — file systems, archives, others — address these topics (and add to them):

1. Change control and configuration management.
    A. What specific configuration management tools, methods, or practices does your center use to validate hardware/software changes and releases to minimize production performance degradation or system downtime?
    B. Can you provide an example of how testing a change/release on a pre-production system provided unique insight into a configuration problem before users detected it?

2. Ongoing system administration.
    A. What specific file system and/or archive metrics does your center measure and monitor on a regular basis, and how have those findings directed which types of operational tasks your center has automated to minimize frequency and impact of production incidents and optimize system performance and end-user experience?
    B. Can you provide an example where self-monitoring and detecting production incidents led to an investigation of root cause to reduce mean time to resolution (MTTR) of future file system or archive outages?

3. Technology refresh.
    A. What unique approach has your center taken to balance the end-user requirement to increase system availability while providing system architects the opportunity and access to the environment to satisfy the ongoing need to refresh underlying file system and archive components?
    B. How does your center expand capacity of either a file system or archive resource while minimizing user impact?

4. Security management.
    A. What strategy is your center taking with, for example, operating system (OS) patching or vulnerability scanning, to satisfy the ongoing and rigorous demands of computer security professionals?
    B. How does your strategy balance the growing need to provide a high degree of collaboration for distributed user communities with access to multiple levels of data sensitivity?
    C. Assuming your center provides a multi-zoned security architecture with various access control levels and technologies, how would you demonstrate to computer security that it is providing adequate protection and controls?

The Reliability and Availability of Storage Systems

For each layer of the storage hierarchy — file systems, archives, others — address these topics (and add to them):

1. Resilient architectures and fault tolerance.
    A. What specific system architectural or configuration practices, decisions or changes (hardware and software) has your center made that have demonstrably improved availability, reliability or performance of your file system or archival system?
    B. Where in the environment should redundant hardware be bought/deployed?

2. Hardware and software maintenance.
    A. What is your philosophy or practice for executing system maintenance down times?
    B. How do those practices contribute to improved system availability and reliability outside of planned outages?

3. Data integrity.
    A. What strategies do you use to ensure data integrity and what parts of the end-to-end compute/store/visualize/archive cycles does it cover?
    B. Do you employ end-to-end checksums within the end-to-end cycle and if so where?

4. Off-hours support and availability.
    A. What mechanisms do you have in place to ensure reliable file system and archive operation during off-hours and, in the event of a facility event, such as power loss, chilled water loss, fire alarm, etc.?
    B. What mechanisms are in place to quiesce storage and to protect it?

The Usability of Storage Systems

For each layer of the storage hierarchy — file systems, archives, others — address these topics (and add to them):

1. What are your major usability issues?

2. What applications/tools have you developed or obtained elsewhere and deployed that have made your storage system more effective and useful for end users?
    A. What tools or methods are available to users for I/O related problem diagnosis? Describe experiences where use of diagnostics resulted in improved outcomes. Are the available diagnostic capabilities sufficiently robust? Scalable?
    B. Please help us categorize the applications/tools in use.
    C. Which tools are used by end users, and which primarily by system administrators?

3. Discuss recent challenges in providing I/O service to your user community, and what practices/strategies were used to meet them.
    A. What trends in user requirements resulted in the need to address the challenges?
    B. How well are current solutions meeting the demands, or where are they falling short?
    C. Where might experience at other sites be helpful to your challenges?
    D. Which of your practices outlined above would you suggest for a best practices list?

4. Large data movement: With respect to internal file and storage systems, do sites dedicate specific resources to data movement internally?
    A. What (if any) direction is given to users for moving data around internally?
    B. What tools are available for helping users improve data transfer performance?

5. At what organizational level are users managing data organization, per user, per code, per project, or some larger unit? Are there common approaches or best practices that have been identified which are being leveraged to aid these efforts?

6. How does your site manage health monitoring of I/O services, and how is pertinent information transmitted to users? What feedback do users have on the content and timeliness of the information?

7. What are your user training and documentation practices?

Workshop Findings

Workshop participants identified the best practices provided in this section of the report as applicable to file systems and archival storage systems operations at high performance computing facilities. The practices summarize the experiences of 29 different sites across many different funding and parent agencies, and represent decades of experience in operations.

In this section, each best practice is accompanied by details to help better understand the meaning of the practice, and implementation examples where practical. Most practices identified have an aspect of usability, reliability, administration and business in operating storage; that is to say that many of the practices were intertwined among breakout sessions at the workshop. For example, reliability investment decisions aren't just about compliance or policies in a particular storage system; they affect the ability to do science and, therefore, the usability of the storage system.

Best Practices

1. **For critical data, multiple copies should be made on multiple species of hardware in geographically separated locations.**
   There is no substitute for multiple copies on dissimilar hardware and software and media formats. Systems that compromise on either hardware, software, or media distinction are more vulnerable to data loss than systems that enable hardware, software, and media diversity (i.e. a single firmware bug can corrupt data stored geographically apart, but on identical hardware/software). Workshop attendees also recommended that sites should have multiple file systems to provide available storage during downtimes and distribute particular workloads to supporting system configurations (e.g., scratch and home, backup and archive). Participants pointed out that backups and disaster recovery plans are classic methods still relevant today of providing the best data protection for critical data and systems retaining critical data.

2. **Build a multidisciplinary data center team.**
   Robust facilities operations are a key component of high performing HPC data systems. Sites agreed that regardless of the specific method, teaming facility personnel with HPC personnel to design, plan, and maintain equipment in support of data systems requirements increased reliability of the data systems. Regularly scheduled meetings involving facility representatives and personnel from every aspect of HPC center operations are critical to maximizing communication and overall HPC center efficiency. Specific facility recommendations for data systems arose at the workshop: having multiple power feeds, priority for use of universal power supplies (UPS), ensuring that power supplies meet high standards (i.e., CBEMA and SEMIF47), and having automation for facility operations (i.e., emergency power-off, startup and shutdown).

3. **Having dedicated maintenance personnel, vendor or internal staff, is important to increasing system availability.**
   By dedicating personnel to specialize on storage hardware and software, the facility will have increased involvement in the issues, needs, and solutions around operating its storage systems. All workshop participants had dedicated storage staff or vendors for their storage systems. By having dedicated personnel, they are able to have the detailed knowledge and experience on hand to prevent many system failures and respond quickly to return the system to operation. Further discussion also pointed out that having development knowledge (e.g., source code, or developers working on storage software) enhanced the reliability and availability in HPC storage systems. Workshop participants pointed to the success of both Lustre and HPSS at the labs involved in

their development as examples of this. Non-development sites often rely on development sites for achieving high availability storage systems.

4. **Establish and maintain hot-spare, testbed, and pre-production environments.**
Data systems in HPC environments are held to high standards of integrity since data is important to the validity of scientific findings. Workshop participants discussed that increased reliability and availability of file systems and archives can be achieved by using hot-spare/burn-in and pre-production systems where software and hardware is tested prior to being placed into production. Attendees identified that the benefits are twofold: to validate procedures (executing both the deployment steps and failback steps) and to validate the readiness of hardware or software in the specific site's environment. Testbed data systems are key to gaining knowledge and expertise of new technologies, and to evaluating future storage hardware and software in isolation from the production environment. Participants identified the importance of not using the pre-production environment for evaluation of hardware or software that is not intended to move into production rapidly, so as not to undermine the stability of the pre-production environment and its ability to closely represent the production environment.

5. **Establish systematic and ongoing procedures to measure and monitor system behavior and performance.**
The complexity of HPC file systems and archival resources presents a challenge to design, deploy, and maintain storage systems that meet specific performance targets. It is important to understand the anticipated workload and use that understanding to design performance benchmarks that reflect that workload. Performance benchmarks and workload simulators should be used systematically from the time new hardware is first under evaluation, through the deployment of a new resource, and regularly throughout the life of the system. Benchmark tests establish a baseline for expectations of the system, and regular testing will reveal most performance degradations due to software and hardware issues, resource contention, or a change in the workload. Share benchmark results with other sites and compare performance with other comparable systems. In addition to performance benchmarks, testing should also include data integrity checks. Monitor usage statistics over time in order to anticipate the growth of demand for resources.

6. **Establish authoritative sources of information.**
In order to answer questions accurately and consistently, there should be a shared and widely known location for information. User documentation, training, and how-to documents should be available and kept up to date. There needs to be a definitive source code control repository for open source infrastructure and benchmark software. Use configuration and change management tools to assist system staff with the administration of the storage resource. Support staff need a central location for procedures and scripts for service disruptions and problem resolution. Make benchmark results, system metrics, and reports on milestones available.

7. **Manage users' consumption of space.**
This was a hot topic in the workshop, where attendees discussed different aspects of managing usage of the storage system. A variety of solutions came forth, involving both active (allocations and quotas) and passive (accounting and auditing) techniques. Most participants used allocations or quotas to manage consumption of storage resources with good success. Participants unanimously agreed that having transparency of capacity to users was important to success in managing a limited resource in high demand. Those that use allocations and quotas noted that they are important to managing data effectively and keeping the system usable. A popular proposal put forth is to consider allocation renewals as a way to revalidate the need for data

retention. Attendees agreed that automation of storage policies is desired (e.g., information lifecycle management, or ILM, capabilities).

8. **Storage systems should function independently from compute systems.**
   Sites that design their file systems, even local scratch, to be independent of their compute system hardware and software improve the availability and reliability of the storage system. Without independence, issues with either the storage or the compute system usually affect each other. There is an added benefit to users in having the systems decoupled: the compute system's data may still be available to users during compute system downtime. For instance, data analysis or visualization could proceed if the file system were available to platforms capable of such work. It was further pointed out that the file and archive systems should be able to be standalone as well. Integrated file and archive systems that are so tightly coupled as to prevent usage of either the file or archive system in the event the other is unavailable are not recommended. Workshop participants identified that this practice is achievable with either dedicated file systems (e.g., local scratch file systems) and center-wide or global file systems. Though workshop participants agreed on this practice, they noted that facilities should also ensure they provide mechanisms to pre-stage or move data between the distinct systems for greater usability.

9. **Include middleware benchmarks and performance criteria in system procurement contracts.**
   Users have an expectation of predictable performance. The importance of storage to HPC systems needs to be stressed from the beginning of the procurement process. A fraction of the overall budget, in the range of 10% to 20%, should be dedicated to file system and storage resources in keeping with the goal of achieving target performance. Performance targets should include middleware libraries, and DOE sites should fund performance enhancements to those libraries. The desired result is that users see "portable performance" between sites when using middleware libraries that have been transparently optimized for the local storage architecture.

10. **Deploy and support tools to enable users to more easily achieve the full capabilities of file systems and archives.**
    Often users express frustration with achieving expected or reported performance of file systems and archives. Workshop participants shared a number of different tools that aid a user in exploiting the performance of the file system or archive. Specifically, the workshop identified the following tools as particularly useful:
    - Parallel Storage Interface (PSI), HTAR, and spdcp for parallelizing metadata and/or data operations
    - GLEAN, ADIOS, and Parallel Log-structured File System (PLFS) for restructuring I/O to better match underlying file system configuration
    - Darshan, Integrated Performance Monitoring (IPM), and Lustre Monitoring Tool (LMT) for providing monitoring and diagnostic information to troubleshooting poor I/O performance
    - Hopper, EMSL, NEWT, and GlobusOnline for providing web interfaces to improve access and operations on data.

11. **Training, onsite or online, and meeting with the users directly improves the utilization and performance of the storage system.**
    Training and educating users on storage system operation (e.g., its capabilities and limitations, architecture, and configuration) and recommended usage (e.g., special options or flags to consider or to avoid) benefits the utilization, administration, and user satisfaction with the storage system. Most workshop participants had web pages and various other forms of usage and system documentation; however, some sites had online videos providing details about using their storage

systems, and others conducted onsite training or worked one-on-one with their users. All agreed that the more engaged a site is in providing user training on storage systems, the more satisfied and positive the users of storage tended to be. Several of the breakout sessions proposed that transparency of storage system operations and capabilities is key to managing user expectations and important to operating a successful storage system. Training and documentation are the most successful ways of providing transparent storage system operations.

12. **Actively manage relationships with the storage industry and storage community.**
Attendees agreed that sites should avoid objective metrics and penalties in acquiring and maintaining storage systems, instead working towards partnerships between customer and industry. Some examples of collaborations in storage are HPSS and Lustre (e.g. OpenSFS). In situations without strong collaboration, it was noted that user groups and working groups are an effective way for the storage community and stakeholders to work towards storage system improvement to mutual benefit between the participants. Participants also recommended encouraging diversity in HPC storage solutions to prevent risk from product extinctions. By developing a relationship with the storage industry, sites better understand upcoming technologies and products to improve file or archive system operation. Many sites advocated supporting open source technologies to mitigate risk. All participants recommended stronger communication and collaboration amongst fellow HPC storage administrators as important to improving file and archive system operations.

13. **Monitor and exploit emerging storage technologies.**
To the extent possible, HPC systems benefit when they can exploit commodity products and services. As new technologies emerge, DOE centers need to examine them and determine if and how those technologies fit in the HPC environment. Cloud services, for example, may provide cost benefits in high volume data processing, but may also require scientists to rethink their application architectures. An early and careful evaluation of emerging technologies can mitigate the risk of making a choice with a poor long-term outcome. As the cost and performance of flash and spinning media evolve, the underlying architecture may need to change; for example, the storage hierarchy may replace tape with disk.

14. **Continue to re-evaluate the real characteristics of your I/O workload.**
While data intensive computing is creating I/O workloads that exceed Moore's law, there is a growing gap between computation and storage capabilities. Monitoring tools like Darshan and the Lustre Monitoring Tool (LMT) can help capture changes in user requirements and the I/O workload on currently deployed systems. As the science being conducted evolves, this will lead to changes in future requirements. Those changes must inform the procurement and planning of systems. The underlying science is driving an evolving set of storage requirements both for bandwidth and volume. It is becoming important to track storage metrics they way the HPC community tracks flops and cycles.

15. **Use quantitative data to demonstrate the importance of storage to achieving scientific results.**
Workshop attendees felt there was an element of advocacy in establishing clear and detailed storage system requirements. An "I/O blueprint" can quantify both the costs and benefits of required storage. Since an under-resourced storage hierarchy will impact the amount of science that can be accomplished, that negative impact represents an opportunity cost. A comparison between that opportunity cost and the storage system expense can allow the HPC system to be designed to have storage that balances it.

16. **Retain original logs generated by systems and software.**
    Workshop participants noted that pristine logs are critical to security forensics, problem diagnosis, and event correlation.

While identifying the current best practices for operation of HPC storage systems, we spent time also discussing changes in user needs based on scaling our systems to the exascale level, which implies exabyte file systems and archives. The current gaps and some future recommendations to prepare file and archive systems for the exascale era of storage are provided in the next section.

Gaps and Recommendations

The information that follows is divided into two sections. First are gaps that exist in current operations of HPC storage systems for which solutions do not exist, but which the community or industry is progressing towards. Second are gaps that are not being addressed operationally and that demand revolutionary approaches to solve.

Gaps likely addressed by evolutionary solutions:

1. **Incomplete attention to end-to-end data integrity.**
   While no HPC solution exists today providing end-to-end (client to system and back to client) data integrity, the community contributes and supports the T10PI standard. The standard is currently being adopted for components used in the HPC environment.

2. **Storage system software is not resilient enough.**
   An otherwise normal hardware error can cause a storage system outage due to current software limitations. Today this impacts the choice of hardware and the amount of facilities infrastructure required to sustain reliable storage systems.

3. **Redundant Array of Independent Tape (RAIT) does not exist today.**
   The only data protection that exists today for tape is multiple copies. However, the HPSS collaboration is working on an implementation of RAIT for use within HPSS.

4. **Provide monitoring and diagnostic tools that allow users to understand file system configuration and performance characteristics and troubleshoot poor I/O performance.**
   Today there exist several tools for monitoring file system performance: Darshan, IPM, and LMT. However, these do not have broad deployment across HPC sites and need further support for code maintenance and feature improvement to be more widely accepted.

5. **Communication between storage systems users and storage system experts needs improvement.**
   There are few storage experts at HPC sites, and it is rare to identify individuals on user projects who are responsible for focusing on storage or I/O. Further, there are rarely consultants at HPC centers who focus on or have the skill set to manage storage or I/O issues. As storage and I/O increasingly become focal issues in HPC projects, this will improve. Sites are working on identifying efficient and effective user education mechanisms (e.g., online videos, onsite workshops).

6. **Need tools and mechanisms to capture provenance and provide lifecycle management for data.**

Workshop participants identified a growing focus on provenance and lifecycle management as new policies that mandate them are being developed under the America COMPETES Act, which invests in science and technology to improve the United States' competitiveness.

7. **Ensure storage systems are prepared to support data-intensive computing and new workflows.**
   Storage systems are designed for computational system needs today. However, new instruments such as genomic sequencers and next generation light sources have tremendous bandwidth and capacity requirements alone that will strain existing systems. As well, the push towards data-intensive computing will result in different system architectures than exist at most HPC facilities today.

8. **Measure and track trends in storage system usage to the same degree we measure and track flops and cycles on computational systems.**
   This will improve understanding of the capabilities and limitations of the system in use, and will help with improving quality of service for users in a shared storage environment.

9. **Tools to provide scalable performance in interacting with files in the storage system.**
   This is focused on tools users require to interact with files (cp, tar, gzip, grep, etc.). Tools help, but there is room for improvement beyond tools with existing systems.

Gaps requiring revolutionary approaches:

1. **Need quality of service for users in a shared storage environment.**
   Nearly all sites represented at the workshop have shared or centralized storage environments. Attendees recognized the need to provide a higher level of performance to users that need it. In current shared environments, all users experience the aggregate performance of the shared storage system. Unlike compute resource managers, storage managers have no method of scheduling workloads or even understanding how to prevent competing I/O workloads from affecting each other.

2. **Need standard metadata for users to specify data importance and retention.**
   One common example is that it is impossible today to specify the lifetime of a file as an attribute that stays with the file through whatever number of storage systems the file moves through. This is required for management of the data.

3. **The diagnostic information currently available in today's storage systems is woefully inadequate.**
   With any storage system in HPC use today, it is still difficult and time consuming to figure out what user is affected by or causing a particular problem. This is primarily because HPC systems in use today are all distributed and have large and complex architecture. For example, the scale of systems today makes seemingly simple operations, such as finding a user (bad actor) who is willfully or unknowingly impacting center performance, impossible in real time.

4. **Metadata performance in storage systems already limits usability and won't meet the needs of exascale.**
   There are two main problems with increasing metadata performance in storage systems today: software design for distribution of metadata operations, and reliable hardware limitations for accelerating metadata operations. Storage system software requires design to enable parallel and distributed metadata operations to enable the best performance. Metadata is normally a relatively

small amount of data in even very large storage systems today. Finding a reliable but increasingly fast small storage device that is affordable is challenging. In addition, improving the usefulness of metadata (e.g. indexes, extended attributes) to users of storage systems means having more small fast reliable storage devices.

5. **POSIX isn't expressive enough for the breadth of application I/O patterns.**
   POSIX compliance limits the ability to achieve maximum performance in storage systems. To achieve improved performance, most storage systems relax POSIX requirements (e.g. lazy updates to file timestamps). A new approach is needed to enable a broader range of application I/O without burdening the application with the complexity of keeping their data consistent.

6. **There exists a storage technology gap for exascale.**
   Storage technology is still improving at a slower rate than compute or networking technology. New forms of storage, namely solid state, will help but not solve the problem. Probe memory is one of the most promising in terms of performance characteristics that could significantly boost storage system capability, but it isn't expected until at least 2015, and it's only being worked by a select group of storage vendors.

Appendix A: Position Papers

The papers included in this section were the position papers requested of and submitted by workshop attendees. They represent a collection of ideas, architectures, and implementations surrounding the practice of running production storage systems. The papers facilitated discussion and identification of Best Practices for operation of file and archival storage systems.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

First Author Name	**Second Author Name**
Affiliation	Affiliation
e-mail address	e-mail address

Philippe DENIEL
CEA/DAM
philippe.deniel@cea.fr

ABSTRACT / SUmmary

CEA/DAM manages two compute centers : TERA100 (first SC in Europe) which is dedicated to classified applications and TGCC which is an open compute center for institutional collaboration (see http://www-hpc.cea.fr/en/ for details). The management of produced data lead CEA's teams to deal with several specific issues, making them develop their own solutions and tools. This paper is focusing on fFiles's Llifetime, and metadata management.

INTRODUCTION

CEA/DAM has been involved in HPC for many years. Because the compute has widely increased, the amount of produced data as drastically increased as well, making it necessary to have dedicated systems and dedicated teams to handle the architecture in charge of storing the data. This situation leads to several challenges : keeping data available to end users is of course one of them, but not the only one. With a huge amount of data comes a huge amount of metadata records. Consideration haves to be taken to manage them. Last, the data kept areis not all of same value. When some files are criticals, others are not, but managing this aspect may be painful to the user who has thousands of files to delal with and sort. Tools have then to be made available to users to help them deal with information life cycle.

Quotas and retentions

Ian Fleming said “diamonds are forever”, but for such files are not forever. The main issue there comes from the users. They produce lots of data (a daily production of 30 to 100 TB a day is a very common situation at CEA/DAM), but they often done't care about what the data become. This leads to a perpetually growing storage system where less than 1% of the content is accessed. Finally a big amount of files will never be read and are even totally useless once the run of the code is over (checkpoint/restart files for example). But the truth is this : if not forced, a user will never delete his files. Two main reasons for this:

- Llack of time
- Afraid fear of accidentally deleteing useful data

I suggest two solutions to handle this. The first is an old-fashioned Unix paradigm : quotas. The second is more sophisticated and is based on extended attributes to implement files's retentions.

Quotas usually works on a “space used” and a “used inodes” basis. File's size is not that critical (modern FS and storage system are huge today), but consideration on inodes are more interesting because they depict well the numbers of metadata records owned by a user. This is interesting in today's situation where the meteadata footprint becomes the filesystem's limitation. Quotas are simple to set, manage and query (quotactl function in the libC, RQUOTAv2 protocol to be used jointly with NFS), but is has its inconvenientlimitations. One of them is the distributed nature of the filesystem used in the HPC world. In a massively distributed product where data areis spread across multiple data servers with parallel pattern, it becomes hard to efficiently keep a centralized place to keep user's information on quotas. Anyway, I suggest that when available, quotas are to be used because they are a simple way of setting limits to the users, making them aware of the amount of files and data that they own.

Files rRetentions is a an other promising another way. The idea is to associate a specific metadata record to every file and directory. This is done by using extended attributes (aka xattr), which makes the assumption that the underlying storage system's namespace handles such a feature. This xattr will contain an information on the object's lifetime. This can be something like “this file will stop being of interest after a given date” or “this file can be considered useless isf not read/written during a defined period”. The key there is to have this metadata for every file (with users input). Specific tools will then audit the file system, produce a list of files to be deleted based on “retention policies”. The user will be warned (mail...) when some of theirhis files are candidates for deletion. Finally files are purged. This approach can lead to a virtuous circle : when producing data, users will take the habit to set the parameters to tell how long they'll requireneed the files, giving to the administrator input on their file's lifetime. This is good for the sysadm that who will save space on his storage system, and this is good for the user can who can schedule the deletion of his files, avoiding the painful task of cleaning his directories when quota limit is reached.

Metadata management

Past challenges tofor filesystems wasere size : would the available resources be large enough to store everything I want to put in the system ? Then come performance consideration, and the idea that the users hate to wait to access their data. Right now, these aspects are addressed by modern filesystems (for example Lustre which is widely used at CEA) that are based on a distributed design relying on multiples data servers.

But many files means many metadata records and this can quickly be problematic, especially in a HPC environment. People who have once seen a single directory with hundreds of thousands of files in it know what I am speaking about. Beyond the technical consideration (big directories are an “edge” situation), the manageability of such exotic objects is a real problem : a single “ls -l” in it may last for hours.

Frequent filesystem audits (like those from CEA's RobinHood product (http://robinhood.sf.net)) helps in this : it becomes easy to identify “nasty” patterns in users directories and takes corrective actions. For example, the admin could decide to pack a big directory into a single tar file. Providing users with tools withusing “best practices enforcement” is also a way we follow. Copying Data copydata to the storage system goes is to delegated to a utilitytool that can decide to pack the data automatically.

Metadata volume is definitely an aspect to be seriously considered. Data volume issues have been solved by striping the data. It may not be so easy to stripe metadata because they carry internal dependencies (a file belongs to a directory and can exist with several names if hard links are available) which may limit the algorithms. I actually believe that the main challenge for the filesystems on exascale compute center will be metadata management. Starting into considering this issue today, by setting limits to users to prevent them for to creating “file systems's monsters” and by teaching them the good practices is definitely something to be done today.

ConclusionS

The Exascale systems are coming tomorrow. Beyond the compute power's revolution, there is an incredible technical gap for the storage system. Data management will not be the greatest challenge, but metadata management will. The systems we will have at this time will store data that are produced today or have been generated in the past years. If we are not careful today, we will come to an excruciating situation in the future. And for sure, tomorrow's issues can be smoothed today by setting metadata's useage limits (quotas, retentions) and by providing users with tools to reduce metadata production.

I/O Performance Measuring – White box test and Black box test-

U.S. Department of Energy Best Practices Workshop on File Systems & Archives

San Francisco, CA September 26-27, 2011 Position Paper

FUJITA, Naoyuki
JAXA: Japan Aerospace Exploration Agency
fujita@chofu.jaxa.jp

SOMEYA, Kazuhiro
JAXA: Japan Aerospace Exploration Agency
someya.kazuhiro@jaxa.jp

ABSTRACT

The complexity of the component of file system/storage system (Thereafter, called the system.) is given to one of the reasons that the I/O performance measurement doesn't generalize. Here, let's think about the scene that discusses the I/O performance. Two cases are greatly thought. One is characteristic grasp of the system, and another is comparison between systems. We insist on using the white box test and the black box test properly in this paper. It is necessary to understand a detailed characteristic of the system by the white box test to guide an appropriate I/O operation to the system user and for the I/O tuning. On the other hand, when other systems and one system are compared, you should start from black box test comparison for a constructive discussion because of each system has each design and architecture.

INTRODUCTION

CPU benchmarking is widely discussed and some major benchmark suites[1,2] exist. However, I/O benchmarking is not more general than that of CPU. Therefore, generally speaking, I/O performance measuring and discussion are difficult. In this paper, we insist on using the white box test and the black box test properly.

As you know, white box test is a test done under the design and architecture of the system is understood. On the other hand, black box test is a test done without requiring them. In case of this time, design and architecture is a component, and the connection relationship of the system such as file systems and the storage devices.

WHITE BOX TEST EXAMPLE

This chapter shows examples of the white box test strategy and result. One result is a system that was operating in 2000(Thereafter, called 2000 System), another one is a system that has been operated since 2010(Therefore called 2010 System).

White box test strategy

As a number of nodes increases in HPC system, the system design and architecture becomes complex and changes it's characteristic. We propose layered benchmark as white box test, and show some results. Layered means device level(Measuring Point 1), local file system level(Measuring Point 2), network file system level(Measuring Point 3), and FORTRAN level(Measuring Point 4). Fig. 1 shows measuring points on recent HPC System.

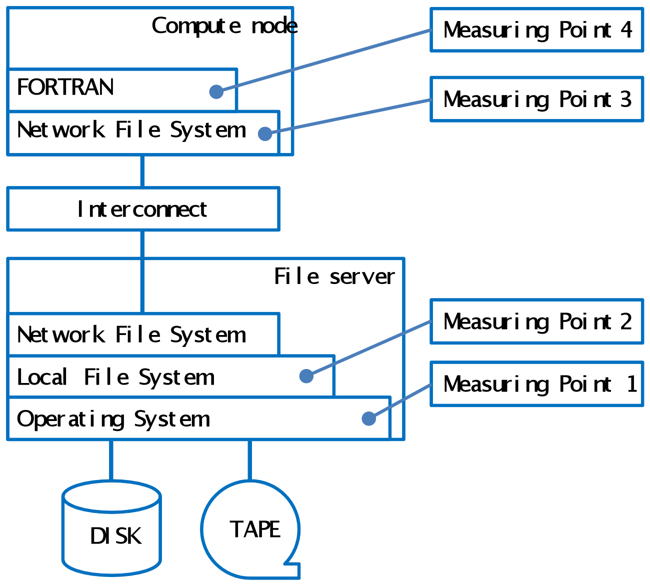
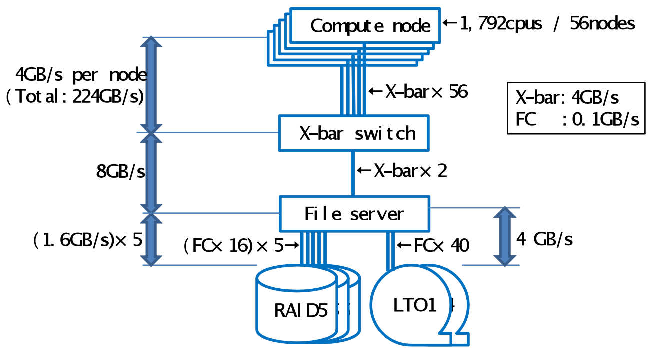


Fig. 1 HPC System I/O measuring points

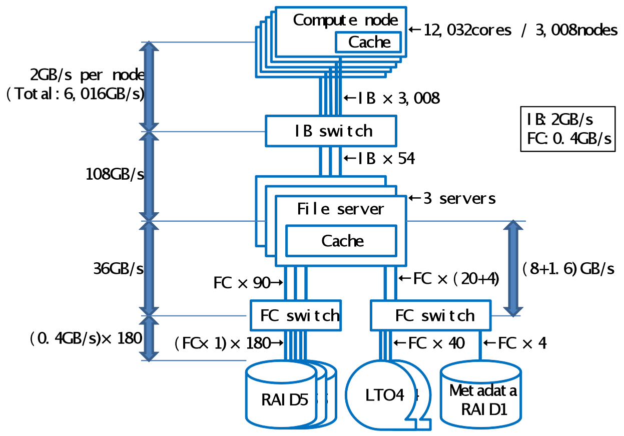
White box test results

Fig. 2 shows 2000 and 2010 system configuration chart. Each system has Compute nodes, Interconnect, which are X-bar switch and IB switch, and File server(s). So there is no change in a basic composition. As a partial change point, 2010 System has clustered file servers and storage devices are attached via FC-SAN switches. Fig. 3 shows the result of the layered benchmark of these two systems. There are some bottleneck points in a system. To analyze a bottleneck, we aim at file system cache, interconnect bandwidth, and DAS/SAN bandwidth and its connection relationship design.

Device level benchmark showed similar result between two designs except disk write performance. But the characteristic of network file system level was very different. In this case, it depends on file system cache effect.

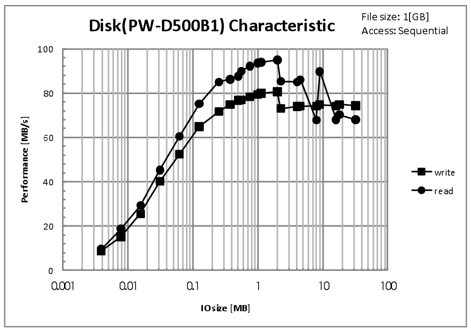


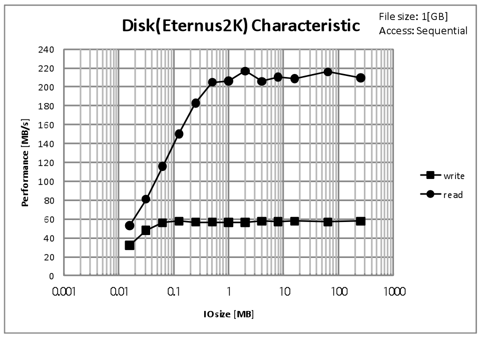
(a) 2000 System

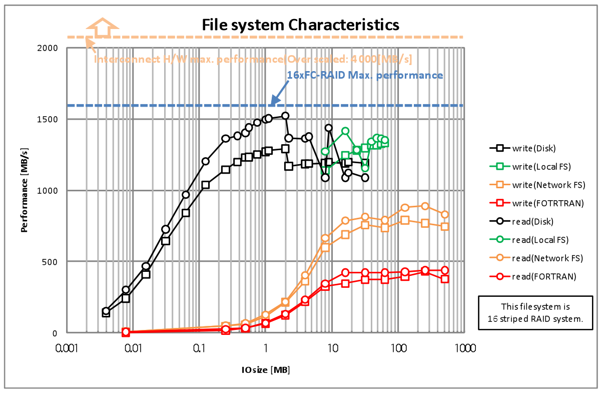


(b) 2010 System

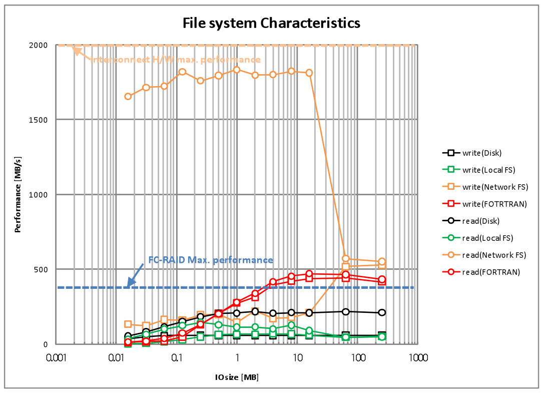
Fig. 2 File system and storage system configuration







(a) 2000 System



(b) 2010 System

Fig. 3 Layered benchmark results

BLACK BOX TEST EXAMPLE

This chapter shows examples of the black box test strategy and result.

Black box test strategy

As we said, each system has each design and architecture, so the comparison of simple I/O performance is not significant. But when we discuss about I/O performance, especially compare with several systems, first of all, it should take a general view of a rough performance. A common tool to measure the file system performance that is appropriate for the measurement of large-scale storage doesn't exist, and the performance measurement tool is made individually in each system and the performance is evaluated individually. In addition, as a peculiar operation to the file system will be needed, it is difficult to compare it with the performance measurement result in another file system. Then, we model the measurement tool and the measurement item, and propose the method of simply diagnosing the characteristic of the large-scale storage system based on the result of a measurement that uses the tool[3].

Objective

It aims at the thing that the following two points can be measured generally in a short time.

(1)Checkup of installed system

Whether the performance at which it aimed when the system administrator installed the system has gone out is examined.

(2) Routine physical examination under operation

Grasp of performance in aspect of user. The operation performance is measured.

Diagnostic model

(1)Checkup of installed system

-Maximum I/O bandwidth performance

Read performance immediately after Write. It is assumed that data are in the client cash.

-Minimum I/O bandwidth performance

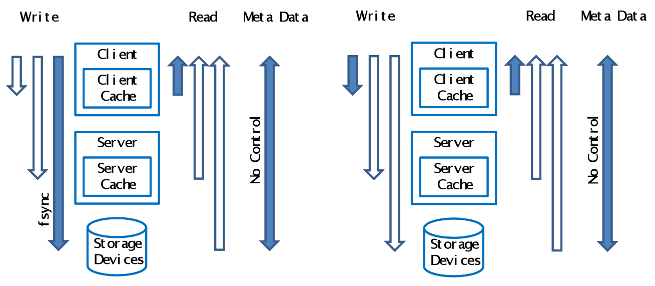
Fsync is assumed after Write and the multiplication cash assumes all things forwarded to the real storage device.

-Meta data access performance

The presence of cash is not considered (Because the cache management cannot be controlled in the black box test).

(2) Routine physical examination under operation

This diagnosis tool is regularly made to work while really operating it, and the state grasp is enabled. In this case, it is assumed to gather the maximum performance (cash hit performance) from the viewpoint of the user aspect. An enough prior confirmation by the system administrator is necessary to make the measurement tool work regularly. Moreover, customizing the measurement tool (measurement downsizing etc.) might be needed. The measurement model is shown in Fig. 4.



(1)Checkup of installation (2)Routine examination

Fig. 4 Measurement model

Measurement tool and item

Using IOR.

(a)Large-scale data transfer (Throughput performance: Constant amount of file for each process, large I/O length)

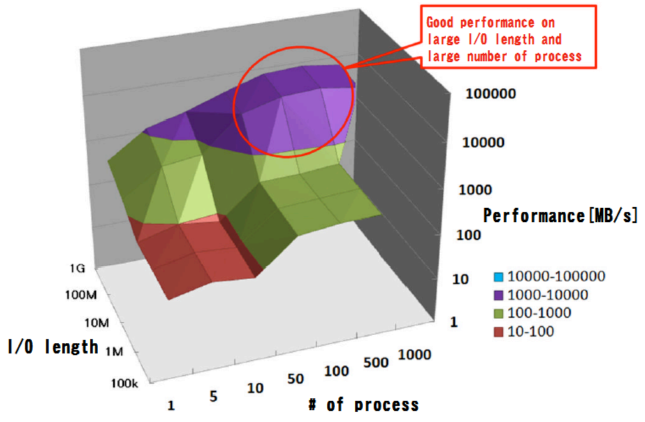
(b) Constant volume of data (Throughput performance: Small file size, small I/O length)

Using mdtest.

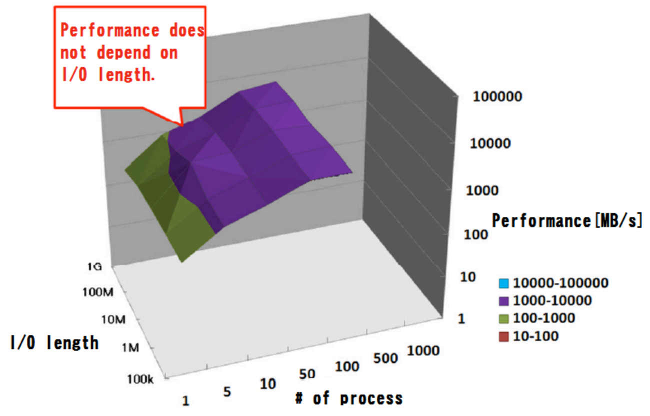
(3) Meta data access (response performance)

Black box test results

Fig. 5 shows the result example of large-scale data transfer on System A and B.



(a) System A



(b) System B

Fig. 5 Large-scale data transfer results

CONCLUSIONS

"What should be measured?"

Each layer benchmark should be done when we want to know the characteristics of the system. The Layers are device level, local file system level, network file system level, and FORTRAN level. This kind of measurement will be done as a white box test.

Modeled measurement item and tool should be used, when we want to compare several systems. The result is a starting point of the discussion. This kind of measurement will be called black box test.

Both white box test and black box test should be used when we manage file system and storage system.

REFERENCES

1. Standard Performance Evaluation Corporation http://www.spec.org/benchmarks.html
2. TOP500 supercomputer Sites, http://www.top500.org/

   Scientific System Society, "Large-scale storage system working group report," 2011,(in Japanese)

Title: U.S. Department of Energy Best Practices Workshop on File Systems & Archives Position Paper

Author(s): Torres, Aaron
Scott, Cody

Intended for: U.S. Department of Energy Best Practices Workshop on File Systems & Archives, 2011-09-26/2011-09-27 (San Francisco, California, United States)

U.S. Department of Energy Best Practices Workshop on File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Aaron Torres
Los Alamos National Laboratory, HPC-3
agtorre@lanl.gov

Cody Scott
Los Alamos National Laboratory, HPC-3
cscott@lanl.gov

ABSTRACT / SUMMARY

As HPC archival storage needs continue to grow, we have started to look at strategies to incorporate cheaper, denser, and faster disk as a larger part of the archival storage hierarchy. The archive in Los Alamos National Laboratory's Turquoise open collaboration network has always used a generous amount of both fast and slow disk in addition to tape. Lessons learned during Road Runner Open Science pointed to the need for large amounts of cheaper, slower disk for storage of small to medium sized files and faster disk in order to store and quickly retrieve file metadata. New advances in the last year may signal another transition that altogether eliminates the need for migrating small and medium files to tape. Improvements in disk speed, particularly solid state devices (SSDs), also allow us to operate on billions of files in a reasonable span of time even as archives continue to grow.

INTRODUCTION

The Open Science simulations run on the Road Runner supercomputer at Los Alamos National Laboratory (LANL) in 2009 provided the opportunity to test an archive based on commercial off the shelf (COTS) components. For this archive, we chose the General Parallel File System (GPFS) and Tivoli Storage Manager (TSM) due to robust metadata features, fast data movement, flexible storage pool hierarchy and migration, and support for a multitude of disk and tape options [1].

This archive joins a long history of archival storage at LANL, including the Central File System (CFS) and High Performance Storage System (HPSS). Thanks to administrative diligence, we have or can recreate records about usage patterns of these archives. One similarity we keep seeing in large archives, with the COTS archive being no exception, is that we primarily store numerous small to medium sized files rather than storing large to huge files. As of May, 2011, HPSS at LANL houses nearly 163 million files with total size of 19.6 PB with an average file size of 131.5MB [2]. NERSC publishes similar statistics with an archive housing over 118 million files and 12 PB for an average size of 109MB [3].

ROAD RUNNER LESSONS LEARNED

When designing for archival storage, one often considers the extreme case for file size. In HPC this generally means designing for enormous files on the order of terabytes for current supercomputer sizes. In practice, however, we see a tremendous amount of small to medium files, especially with users performing n-to-n writes or using the Parallel Log-structured File System (PLFS) to effectively convert n-to-1 writes to n-to-n [4]. In the case of Roadrunner, 20 million 8-16 MB files were archived in one weekend [5].

For the COTS archive, this proved to be the largest pain point since the Hierarchical Storage

Manager (HSM) feature of TSM does not currently support aggregating smaller files together when moving them to tape, resulting in poor performance. Users could aggregate their own files using the "tar" command, but they cannot be relied on to do this for all cases. Another option would be to put file aggregation into an archive copy tool such as how the LANL-developed Parallel Storage Interface (PSI) does with the Gleicher developed HTAR [6]. However, doing so breaks POSIX compliance because no other standard file system tool can read or write files aggregated in this way. One of the design goals of the COTS archive was to leverage as much standard software as possible.

For the COTS archive, moving small files to tape without a transparent file aggregation technique did not make sense. So, small files are kept on RAID 6 disk arrays and backed up to tape. RAID provides recovery from minor amounts of single disk failure, and the tape backup provides disaster recovery. Moreover, TSM's backup function does support aggregating small files before sending them to tape.

The COTS archive has 122 TB of fast fiber channel disk to act as a landing area for new files and 273 TB of SATA disk for files under 8 MB to be moved to. Finally, it has 3 PB of tape for files over 8 MB and for the backups of the SATA disk pools. Currently, the archive houses over 107 million files with a total size of 2.1 PB and an average size of 21.22 MB according to our latest statistics as of August, 2011. As shown in Figure 1, 97 million files are less than or equal to 8 MB. This indicates that the general case for our archive is large amounts of smaller files.

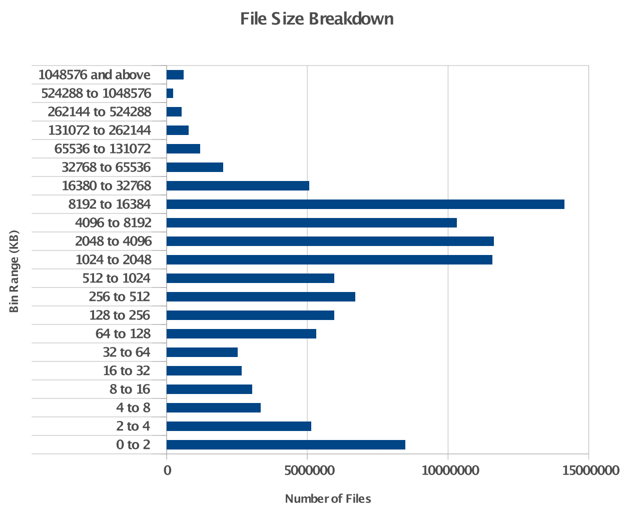


Figure 1. File Size Breakdown of COTS Archive.

RECENT ADVANCEMENTS

The recent explosion of "cloud" backup providers like Mozy, Backblaze, and others lead to questions about how we store large amounts data and if we are doing it in the most cost effective way. For a cloud-based backup service, density and uptime are the two primary driving forces because users continue to back up ever larger amounts of data as they put more of their life on the computer in terms of photos, videos, etc. and data may be backed up or restored at any time. These are also motivating factors for HPC archives. On September 1st, 2009, Backblaze posted an entry to their company blog describing their Backblaze Pod capable of storing 67 TB of data in a 4U enclosure using 47 one terabyte drives for $7,867, or 11.4¢ per gigabyte [7]. On July 20th, 2011, they posted an updated entry now indicating that they can store 135TB in 4U using 47 three terabyte drives for $7,384 or 5.3¢ per gigabyte [8]. Also, Backblaze notes they have deployed 16 PB of disk in the last 3 years [8]. In terms of raw storage, that is within striking distance of the size of LANL's largest HPSS archive at nearly 20 PB.

On the other end of the spectrum, eBay recently replaced 100 TB of SAS disk with SSD [9]. They did this to speed up virtualization and reduce the size of their disk farm. They had a 50% reduction in standard storage rack space and a 78% drop in power consumption by moving to SSD. Although it is impossible for an HPC archive to take this approach, it is possible to replace portions of the total system for tremendous benefits.

An example of using SSD in a storage hierarchy is IBM's recent efforts at speeding up GPFS using SSD [10]. By storing GPFS metadata on SSD, IBM saw a 37 times speed improvement for metadata operations and was able to scan 10 billion files in 43 minutes. For comparison, it takes roughly 20 minutes to scan the 120 million files on the LANL COTS archive using eight 15,000 RPM fiber channel disks in four RAID 1 stripes.

CHEAP, DENSE DISK

The growth of density in hard disks shows little sign of slowing down. In the 2 years between Backblaze posting blog entries about their Pod system, the cost of the Pod actually went down even though the raw capacity of the 4U box doubled. Hard disks in the 4 TB range are on the horizon for desktops and servers in the next year [11], and even laptops are moving to 1 TB disks [12]. The Hitachi 3 TB drives used by Backblaze can be purchased for $120-130 from a variety of retailers [13]. For comparison, an LTO5 tape that holds 1.5 TB of uncompressed data costs approximately $60 [14]. For the same capacity, the disk costs as much as the tape.

One interesting move by companies like Backblaze is that they use consumer level hard drives instead of "enterprise ready" drives. Such drives are substantially cheaper; with the enterprise version of the Hitachi 3TB drives costing over $320-350 per drive as of August 23, 2011 [15]. Backblaze also takes advantage of the manufacturer's 3 year warranty to get a replacement disk if one fails rather than an expensive maintenance contract. HPC archives might be able to leverage the same kind of disk drive by taking into account the disk failure protection afforded by RAID 6 and by having a tape backup of whatever is stored on such disks.

Unlike Backblaze and their Pod, we do not want to be in the custom hardware business. So, we looked for existing commercial hardware that could get the same density of disk. We found the SuperMicro SuperChassis 847E26-RJBOD1 [16]. It is a storage chassis that can support 45 disk drives in 4U. It does not have a built-in motherboard like the Backblaze Pod to manage the disk, but the COTS archive already has machines in its GPFS cluster that can easily take a RAID card with an external SAS connector to plug into this storage expansion chassis. Filling the chassis with 3 TB consumer level disk drives and including the RAID card costs approximately $12,000, or 8¢ per gigabyte.

The idea behind these enormous disk pools is not to completely replace tape, but to adjust the size of file that gets moved to tape. It is entirely feasible today to change the threshold used in the COTS archive from 8 MB to 1 GB with the current price of disk. With this change, we can move all files 1 GB and larger to tape. This file size is also much closer to the size of file necessary to get a tape drive up to peak streaming speed based on internal testing done at LANL. In addition, backing up any file that will stay resident on a disk greatly reduces the fear of failed tapes when storing enormous quantities of small files.

One argument against disk in archive is that archives are usually "write once and read never," so it does not make sense to "waste" power and cooling on spinning disk for data that may never get read. For the COTS archive, the ability of GPFS to move data to different types of storage (ie, fast disk, slow disk, and tape using TSM) based on arbitrary criteria like dates could be leveraged to move rarely or never read data to tape. The tape performance hit is acceptable because the data is essentially "cold". Similarly, a large disk pool can be used to stage data for reading if a user knows he or she will be pulling some set of data from the archive.

FAST DISK CACHE

As HPC archives ingest ever more data because of exascale supercomputers, the metadata will probably become more and more important. At some point in the not to distant future, users will want to search the archive on metadata instead of being forced to create complex directory hierarchies to find the files they are interested in. An example query could be "find all the checkpoint files that were copied to the archive within the last 3 days." Thus, it is also important to quickly search an archive's metadata, whether it is in a file system like GPFS or a database like HPSS using DB2. Here is where faster disk systems like SSD can be used to great effect in HPC archives. As mentioned previously, IBM's testing of storing GPFS metadata on SSD and being able to scan billions of files in less than an hour shows how such fast disk can be very useful.

Another pilot program at LANL is testing metadata performance on SSD using the GPFS COTS archive as a basis for the number of files and types of files stored. Having the ability to quickly scan the metadata of the entire archive provides many benefits, particularly to future research projects and in data management including ongoing work to index and quickly search archive metadata.

In addition, as SSD storage becomes cheaper and denser, it may eventually be possible to replace our fast disk cache, currently consisting of fiber channel disk, with a large pool of SSD similar to how eBay replaced their SAS disk environment. With our current data requirements this is still cost ineffective, but it is worth examining and testing now for the future.

CONCLUSIONS

There are many advantages to having a large, easy to manage pool of disk. When raw speed is not a requirement of this disk, there are solutions available to procure, maintain, and deploy a tremendous amount of disk cost effectively that compares very favorably to the cost of tape. Taking advantage of faster disk like SSD for metadata and disk cache will also benefit future HPC archives. The LANL COTS archive is in a unique position to test and potentially deploy some of these newer solutions in-place with limited negative effect to users.

REFERENCES

1. Hsing-bung Chen, Grider Gary, Scott Cody, Turley Milton, Torres Aaron, Sanchez Kathy, Bremer John. *Integration Experiences and Performance Studies of A COTS Parallel Archive System.* IEEE Cluster Conference, 2010
2. *Accounting Data.* http://hpss-info.lanl.gov/AcctProcess.php
3. *Storage Trends and Summaries.* http://www.nersc.gov/users/data-and-networking/hpss/storage-statistics/storage-trends/
4. Bent John, Gibson Garth, Grider Gary, McClelland Ben, Nowocznski Paul, Nunez James, Polte Milo, Wignate Meghan. *PLFS: A Checkpoint Filesystem for Parallel Applications*. Super Computing, 2009
5. Scott, Cody. *COTS Archive Lessons Learned & Fast Data Pipe Projects*. JOWOG-34
6. Gleicher, Michael. *HTAR – Introduction.* http://www.mgleicher.us/GEL/htar/
7. Nunfire, Time. *Petabytes on a Budget: How to build cheap cloud storage.* http://blog.backblaze.com/2009/09/01/petabytes-on-a-budget-how-to-build-cheap-cloud-storage/
8. Nufire, Tim. *Petabyes on a Budget v2.0: Revealing More Secrets.* http://blog.backblaze.com/2011/07/20/petabytes-on-a-budget-v2-0revealing-more-secrets/.
9. Mearian, Lucas. *Ebay attacks server virtualization with 100TB of SSD storage.* http://www.computerworld.com/s/article/9218811/EBay_attacks_server_virtualization_with_100TB_of_SSD_storage.
10. Feldman, Michael. *IBM Demos Record-Breaking Parallel File System Performance.* http://www.hpcwire.com/hpcwire/2011-07-22/ibm_demos_record-breaking_parallel_file_system_performance.html
11. Shilov, Anton. *Samsung Shows Off Prototype of 4TB Hard Disk Drive.* http://www.xbitlabs.com/news/storage/display/20110308081634_Samsung_Shows_Off_Prototype_of_4TB_Hard_Disk_Drive.html
12. Altavilla, Dave. *A Terabyte For NotebooksL WD Scorpio Blue 1TB Drive.* http://hothardware.com/Reviews/1TB-WD-Scorpio-Blue-25-HD-QuickTake/
13. *HITACHI Deskstar 0S03230 3TB 5400 RPM 32MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive -Bare Drive.* http://www.newegg.com/Product/Product.aspx?Item=N82E16822145493
14. *IBM – LTO Ultrium 5 – 1.5 TB / 3 TB – storage media.* http://www.amazon.com/IBM-LTO-Ultrium-storage-media/dp/B003HKLHZC
15. *HITACHI Ultrastar 7k3000 HUA723030AKLA640 (OF12456) 3 TB 7200 RPM 64MB Cache SATA 6.0Gb/s 3.5"*

Internal Hard Drive -Bare Drive. http://www.newegg.com/Product/Product.aspx?Item=N82E16822145477

16. *SuperChassis 417E16-RJBOD1.* http://www.supermicro.com/products/chassis/4U/417/SC417E16-RJBOD1.cfm

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Nicholas P. Cardo
National Energy Research Scientific Computing Center
Lawrence Berkeley National Laboratory
cardo@nersc.gov

ABSTRACT

Disk quotas are a useful tool for controlling file system space consumption. However, each file system type provides it's own mechanism for displaying quota usage. Furthermore, each file system could display the information differently. Unifying how quota information is reported would simplify the user's experience.

Also having quotas span multiple file systems would provide users some flexibility in storage usage.

INTRODUCTION

The use, management, and enforcement of disk quotas is often difficult to interpret at the user's level as well as being too rigid of an enforcement mechanism.

Identification of the issues

While disk quotas are extremely useful in managing disk space, they are often complicate, hard to understand, counter productive to the user community.

Lets first examine quota-reporting utilities. For IBM's General Parallel File System (GPFS), the command `mmlsquota` is used

```
nid00011:~> mmlsquota home1
                      Block Limits                               | File Limits
Filesystem type       KB    quota    limit in_doubt grace | files   quota   limit in_doubt grace Remarks
tlhome1     USR 29491548 41943040 41943040    34032  none |  7680 1000000 1000000      429  none fshost
```

to display quota limits and usage.

For Lustre, quota information is obtained with the command `lfs quota`.

```
nid00011:~> lfs quota -u juser /scratch
Disk quotas for user juser (uid 500):
     Filesystem  kbytes   quota   limit   grace   files   quota   limit   grace
       /scratch 2666948       0       0              83       0       0
sc-MDT0000_UUID     120               0              83               0
sc-OST0000_UUID       4               0
…
```

While standard Linux utilizes the quota command.

```
$ quota   Disk quotas for user juser (uid 500):
Filesystem blocks quota limit grace files quota limit grace
/dev/fs0        2   100   200           2    10    20
```

So now there are three different commands each with a different syntax showing different information. Instructing users how to interpret the results can be quite involved. This is especially true when the data is closely examined to exactly what the users really care about. At that level all that matters is what is being consumed and what the limit is. Lustre is quite detailed in its output and provides space consumption information down to the Object Storage Target (OST). This presents a case of information overload as file systems could have 100's of OSTs and each one represents one line of output. But the real question is do users really need to see this.

File system quota reporting also is highly dependant on the file system architecture, and provides details unique to that file system. GPFS provides the `mmrepquota` command producing:

```
                     Block Limits                    |         File Limits
Name  type     KB    quota    limit in_doubt grace | files    quota    limit in_doubt grace
fuser1  USR 17684 41943040 4194304        0  none |    72 1000000 1000000        0  none
fuser2  USR   180 41943040 4194304        0  none |    32 1000000 1000000        0  none
```

Lustre provides no such reporting functionality. Standard Linux provides the `repquota` command for reporting operations.

```
# repquota /quota
*** Report for user quotas on device /dev/fs1
Block grace time: 7days; Inode grace time: 7day
                 Block limits          File limits
User        used soft hard grace used  soft  hard  grace
--------------------------------------------------------
fuser1   -- 1204    0    0          5     0   0
fuser2   --   10  100  200          9    10  20
```

The more file systems types that are present on a system, the bigger the problem becomes.

Along the same lines is that the actual underlying quotas are per file system and cannot be aggregated across multiple file systems. Users must be granted quotas on each individual file system and managed by that file systems quota utilities.

Statement of Position

Quota utilities should be externalized from the products where each vendor is encouraged to contribute to them to support their file system. Furthermore, each vendor should supply a standardized API call to retrieve or manipulate disk quotas. It is recognized that each file system may need to present details not applicable to other file systems. In this case, the utilities should use extended flags to control the operation.

The application of disk quotas needs to be externalized from file system. While the accumulation of accounting data needs to be within each file system, the enforcement of quotas can be externalized. This would allow for a single disk quota to span multiple file systems regardless of file system type. A kernel module could open the quota file, holding the file descriptor open for a system call to access directly from within the file systems.

SUPPORTING DOCUMENTATION

Many quota operations can be easily externalized. Each of the file systems mentioned already provide an API call that can be used to retrieve or manipulate disk quotas. GPFS provides `gpfs_quotactl()`, Lustre provides `llapi_quotactl()`, and standard Linux provides `quotactl()`. This shows that the underlying interface is already in place, but unique to that file system. Linux already can differentiate between the file system types. The mount table contains the field `mnt_type`, which identifies the underlying file system. So why can't a single form of `quotactl()` which utilizes the `mnt_type` to differentiate the file system types be put in place?

The answer is, it can.

User Quota Report

The first utility to make use of this capability essentially replaces `mmlsquota`, `lfs quota`, and `quota`, with a single utility that can display quota information to the users, regardless of file system type.

```
Displaying quota usage for user fuser1:
            -------- Space (GB) --------  ----------- Inode -------------
FileSystem  Usage   Quota  InDoubt Grace   Usage    Quota   InDoubt Grace
---------- ------- ------- ------- ----- -------- -------- -------- -----
scratch          3       -       -     -       83        -        -     -
scratch2        24       -       -     -      334        -        -     -
project          0       -       0     -     1944        -        0     -
common           0       -       0     -       11        -        0     -
u1              28      40       0     -     7680  1000000      429     -
u2               0      40       0     -        2  1000000        0     -
```

In this example, the scratch and scratch2 file systems are Lustre, while project, common, u1, and u2 are GPFS. This utility utilizes `getmntent()` to read the mount table in order to access `mnt_type` which is used to determine the file system type. Then the appropriate `quotactl()` system call is used to access the quota information for the file system. The data is then normalized to a consistent format and presented to the users.

File System Quota Report

Quota reporting at the file system level is very useful for determining the top consumers of the resources. The issue of different file systems reporting different information can be easily overcome. However, the lack of the capability to simply loop through all quota entries a major obstacle had to be overcome. The solution used was to loop on all users to get their usage information. The downside is that if a user is removed from the system and is consuming resources, it will never be reported.

In a similar manner as in the user quota reporting utility, `statfs()` is used to get the `f_type` of the file system. This is then used to determine the correct `quotactl()` to use for that file system. The user list is obtained simply by looping on `getpwent()`.

```
Filesystem: /scratch2
Report Type: Space
Report Date: Wed Sep  7 07:14:37 2011

         ---- Space (GBs) --- Inode ---
Username  Usage  Quota   Usage    Quota
-------- ------ ------ -------- --------
fuser1     8262      0   663560        0
fuser2     7824      0   225937        0
fuser3     5593      0   216674        0
fuser4     4548      0   111542        0
fuser5     2171      0   436872        0
```

The report can be sorted either by space or by inodes. Reported is a simple and easy to read output that is the same regardless of file system type.

Quotas Spanning File Systems

Enforcing file system quotas external to the file system opens up a flexibility to customize the effects when quotas are reached as well as the opportunity to span file systems. Normally quotas are set up with a soft limit that can be exceeded for some grace period while not exceeding a hard limit. The effect of reaching the hard limit is usually the I/O being aborted with the error `EDQUOT` (quota exceeded). Running a large-scale computation for several days that aborts due to quota limits being reached seemed a bit counter productive, not to mention the loss of valuable computational time. Rather than to terminate the run, a better solution would be to allow it to run to completion while preventing further work from starting. A simple check at job submission and another at job startup can prevent new work from being submitted or started without the loss of computational time.

In addition to the flexibility in how to enforce quota limits, the ability to combine usage information from multiple file systems is enabled allowing for a single quota to span file systems. The process is to simply retrieve the utilization from the desired file systems, accumulate it, and then evaluate it. For batch jobs, this can be performed in submit filters or prologues. This is in production at job submission time. If users are over their quota, they will receive a message:

```
ERROR: your current combined scratch space usage of 6 GBs exceeds
your quota limit of 4 GBs.

You are currently exceeding your disk quota limits. You will
not be able to submit batch jobs until you reduce your usage
to comply with your quota limits.
```

This change has improved the users experience on the system while keeping resource consumption in check.

Externally to the file system, an infrastructure was needed to support the ability to grant a quota that applies to all users, as well as exceptions. Some projects simply require more storage resources than is desired to grant to all users. Having a default quota is easy as it is a value that applies to all users. The challenge was the ability to override this while tracking those with extended quotas.

Another utility was created to manage a data file used to track quota extensions.

```
> chquota -R

           --------- Space Quota --------- --------- Inode Quota ------------
Username Q GigaBs Expiration    Ticket       Inodes   Expiration    Ticket      Filesystem
-------- - ------ ---------- ------------- --------- ---------- ------------- ----------
fuser1   U  10240 01/10/2012 110112-000033   5000000 01/10/2012 110112-000033 /scratch
fuser1   U  10240 11/15/2011 110714-000039         - --/--/----       -       /scratch
```

Not only are the new limits for space and inodes recorded, but also the expiration dates for the extension as well as the problem tracking ticket. From a single report, a clear understanding of all existing quota extensions can be ascertained. A feature of this utility is the ability to automatically remove expired quotas. Each not via cron, the command is run to evaluate all quota extensions and remove any that have expired.

Another feature that is targeted to improving the users experience is the ability to inform if a quota extension is about to expire.

```
chquota: your 6 GB space quota on /scr expires on 09/09/11
(110901-000001)
```

The number in the parenthesis is the trouble ticket number tracking the request. This can be placed in login scripts to inform users each time they login to the system.

CONCLUSIONS

Simplifying disk quotas improves usability, reporting, and the user's experience on the system all while controlling consumption of resources.

File system vendors should be encouraged to align their quota implementations into a single command set of tools that provide a consistent interface, regardless of file system type. Until that happens, centers should adopt a plan to develop such tools as they improve the user's experience. Taking this one step further, all centers should adopt the practice of putting these tools into service creating consistency across centers. Many users perform their calculations at several centers and having a consistent set of tools will enhance their ability to work effectively.

By externalizing disk quota enforcement to job submission, users are forced to keep their resource consumption in check without the risk of losing a run due to quota limits. As a result, the computational resources are much more effective as no time is lost due to calculations being cut short when quota limits are hit.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Wayne Hurlbert
Lawrence Berkeley National Laboratory
wehurlbert@lbl.gov

ABSTRACT / SUMMARY

This position paper aims to provide information about techniques used by the Mass Storage Group at the National Energy Research Scientific Computing Center (NERSC) to accomplish technology refresh, system configuration changes, and system maintenance while minimizing impact on users and maximizing system availability and reliability. In particular, it addresses the Center's position that shorter, scheduled outages for archival storage system changes, occurring at familiar times, minimizes the likelihood of unscheduled or extended outages, and so minimizes impact on users.

INTRODUCTION

For the purposes of this discussion, it is taken as given that much of the activity involved in technology refresh, along with configuration changes and other system maintenance, requires systems to be off-line. NERSC's approach to these activities in the archival storage systems, is largely driven by the need to minimize the impact on our users. The notion of minimum impact encompasses the scheduled activity itself, potential fallout from the activity, and the concept of preventive maintenance. This has resulted in a conservative attitude toward system maintenance that favors incremental rather than radical change. In the following I will discuss the motivation and benefits of this approach, and mention some of the real world steps taken at NERSC to implement the approach.

Little by Little

While it can be tempting to "just do it", an incremental approach to technology refresh and other system maintenance activities is usually a viable alternative to the more significant outages often required to accomplish the changes in a single sitting.

Types of projects for which this approach might be helpful include:

- Server upgrades or replacement.
- Significant application, OS, or layered software upgrades.
- Replacement or reconfiguration of disk and tape resources.
- Replacement or reconfiguration of large infrastructure components such as SAN switches.

These projects will often take many hours, sometimes days, to accomplish and run a relatively high risk of unanticipated problems or complications.

An incremental approach indicates that these larger projects be broken up into smaller pieces which can be accomplished in an independent and sequential manner. Naturally, there are projects where this is not possible, for various reasons; our finding is that the reasons are typically not technical in nature.

The Benefits

There are several benefits provided by this approach:

- Less complexity of the tasks executed during an outage, which means a reduction in the likelihood of human mistakes in planning or execution of the tasks.
- Lower risk of aborted or extended outages due to unexpected or unanticipated complications. For example, because fewer tasks are being undertaken, there is a smaller window for hardware failure if devices or servers are being power cycled. Naturally, a device can fail during either an incremental activity or a major project, but the impact on workflow is likely to be smaller, and the impact on the user is likely to be less significant in terms of total time for the outage.
- Easier back out in the case of the need to abort the maintenance activity due to unexpected or unanticipated events.
- Lower likelihood of human error due to the fatigue and stress which usually occur during significant projects.
- When compared with forklift upgrades, lower risk of subsequent fallout due to as yet undiscovered bugs or defects. This is particularly true, obviously, for newer products.
- Where desirable, allows for completing system-down activities during business hours, because of the shorter outages. Business hours may be required in order to insure access to outside expertise.

User Expectations

The incremental approach to performing system maintenance subscribes to the notion that shorter, more frequently scheduled outages will ensure a more stable system, which will better serve users. Outages should be scheduled for a standard day and time, even if not at standard intervals e.g. weekly, with the intent that users will come to expect that time period and plan around it. For instance, on one end of the spectrum, users can simply plan to not run during the normal hours, on the normal day for outages. However, NERSC does provide a programmatic, network based mechanism for automated jobs to check system availability.

Further, NERSC has developed an effective protocol for suspending user storage transfers during short outages. Referred to as “sleepers”, user interface tools on the compute machines look for lock files which cause these clients to loop on the system sleep call until the lock file disappears. The result is that many user jobs simply pause until the outage is completed.

In annual user satisfaction surveys at NERSC, the archival storage resources typically receive high scores with regard to system availability and reliability. [1] [2]

Preventive Maintenance

Preventive maintenance, in the sense of avoiding unscheduled outages and the associated user interruptions, can be seen as primarily concerned with restarting, rebooting, and/or power cycling equipment. These activities usually take relatively little time, and fit nicely with shorter, more frequently scheduled outages. Examples include:

- Reboot to validate configuration changes made while the system is live, even if a reboot/power cycle is not strictly required.
- Reboot to flush out pending hardware failures, or to reset hardware that is in a confused state.
- Rebooting or power cycling also helps maintain familiarity with the way systems and devices behave during power-down and power-up.
- Restarting applications, and less importantly these days restarting operating systems, can

help avoid outages due to software defects such as memory leaks.

- Build rather than copy: when locally built software must be installed on multiple servers, building it on each server validates the installation and configuration of layered software (in addition to allowing debug activities on the various servers).

Example

Project: application upgrade on the current production server hardware, which requires OS and/or layered software upgrades.

The NERSC storage group will typically build a new system disk, from the ground up, on a second disk in the production server.

This will usually involve an outage to install the new OS followed by one to several 2-3 hour outages to install, build, configure, and test (as appropriate) layered software and application code. Each of these outages will involve a reboot to the second system disk for the work to be done, followed by a reboot back to the production disk.

This activity is usually spread out over a number of weeks, and is typically interleaved with other activities that may, or may not, involve preparation for the upgrade.

The upgrade is finalized by rebooting to the new system disk and performing any remaining activities required before going live.

CONCLUSIONS

A conservative approach to system outages for technology refresh, system reconfigurations, and other maintenance can be accomplished through a policy which uses multiple short outages to perform the work incrementally. This promotes greater system stability and minimizes the number of unscheduled outages, resulting in better service to users.

REFERENCES

1. NERSC 2010 High Performance Computing Facility Operational Assessment.
2. NERSC User Surveys. http://www.nersc.gov/news-publications/publications-reports/user-surveys.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives San Francisco, CA September 26-27, 2011 Position Paper

Todd M. Heer
Lawrence Livermore National Laboratory
theer@llnl.gov

ABSTRACT / SUMMARY

Specific to our center, archival data integrity emanates from dual copy files, intensive preproduction environment analysis, and ongoing HSM verification testing. Availability falls from the judicious application of a redundancy model. Efficiency can be obtained by leveraging the large procurements part and parcel of HPC center operations as well as the thinning out of unnecessary costly equipment.

A newly implemented soft quota model constrains growth. Flexibility and communication with users ensure success.

INTRODUCTION

The deployment and administrative tasks of an HPC data archive tax credos of data integrity, service availability, and operational efficiency. Couple that with the specter of prodigious growth, and you have a witches' brew of daunting missions.

DEPLOYING TO OUR CREDOS

I. DATA INTEGRITY

Arguably, the ultimate responsibility of an archive is to protect the data.

- ***Dual Copy, Dual Technology***

Data integrity is achieved by dual copy of files over a specific size range. It is often sufficient to simply have dual copies of a file, unless a specific underlying technology is the source of the problem (e.g. firmware bug causing corruption on a data pattern). In this case, a differing technology must store the second copy.

We dual copy over two tape drive technologies in order to avoid such scenarios. Currently these technologies are Oracle T10000C and IBM LTO-5.

The recent leap in capacity resulting from the barium-ferrite particle of the T10000C media realizes an average of 7.9TB per cartridge with our customer data profile. We have recently been afforded the opportunity to dual copy all files up to 256MB as a consequence. We offer a special class of service customers can specify to obtain dual copy files on tape regardless of size.

Each technology is further separated in two distinct robotic library complexes (Oracle SL8500) separate by a distance of approximately 1 kilomoter.

- ***Offline Testing***

As tape drives are either purchased or replaced due to failure, they are tested for integrity and performance before being placed into production. A suite of tools was created to facilitate this out-of-band testing. Files of known size and composition are written to and read from test media. Timing is conducted and data is examined by means of a checksum. It is important to understand that a performance threshold exists below which drives should be considered faulty for the environment, even if integrity checks pass.

- ***End to end verification***

The largest stride in the quest for complete data integrity can only be realized by testing the entire stack of software and hardware in use by the archive application. We employ a homegrown utility called DIVT – Data Integrity Verification Tool.

DIVT runs as a client on various center platforms while using various source file systems. It transfers files into the archive. The files land on level 0 disk cache. They are then pulled out of the archive and compared against the original. The files are stored again, except this time the file is pushed down to level 1 tape and purged off of level 0 disk. Again it is retrieved from the archive and compared against the original.

Should any anomaly exist, email notification will be sent.

This push and pull against the disk and tape levels of the HSM is constant. Finding problems is a game of percentages. In the last two years, DIVT has found two major problems. The first was a file stat() bug with Lustre parallel filesystem reporting inconsistent file size, the result of which were corrupted tar archive images. The second problem was a tape drive that was silently truncating files, thereby corrupting them on tape. None of these would've been found had it not been for the utility. The opportunity for silent corruption is rampant.

II. AVAILABILITY

The focus is on "nines of availability". Simply stated, it means reducing the length of planned outages. Our goal is often said to be "two and a half nines," or 99.5% annual uptime, which translates into 3.65 hours of outage per month or 1.8 days per calendar year. For this reason, each second of outage is tracked.

- ***Pre-Production***

A "Pre-Production" environment is an absolute necessity to an archive. All new device firmware, device drivers, operating system fixes and version upgrades, and application versions are tested rigorously. It is here where the methods and order of complex integrations take shape. Tuning parameters are also sorted. A substantive subset of the exact hardware used in production should be represented in pre-production.

With such an environment comes the need for discipline. A pre-production system must be fed and cared for in the same way a production system would be, otherwise it quickly achieves a state of neglect, requiring significant resources to restore its usefulness.

We have traditionally run two production environments – unclassified and classified. Each of those has a dedicated pre-production environment. Deployments start in unclassified preproduction. Depending on the nature of the changes, testing can be from a couple weeks to a couple months, after which time it's deemed suitable for production and a planned downtime date is set.

Then the process is started all over for the classified side on its pre-production system. These cycles tend to be much shorter as most software has been battle-hardened in our unclassified environment by this time.

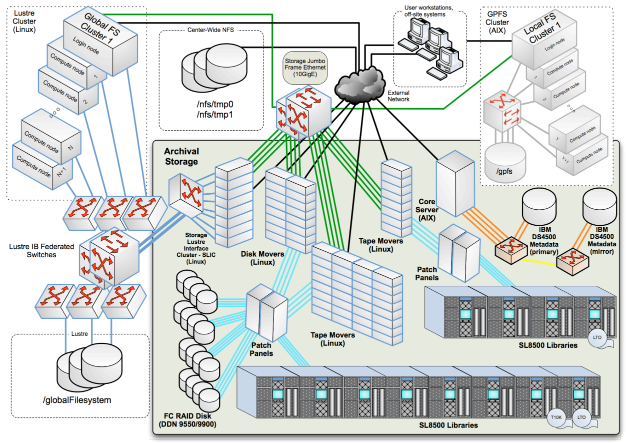
More typically, due to its larger scale, unclassified production will uproot a bug that wasn't caught in preproduction testing. All future deployments are put on hold while problems are researched and remedied.

The net result is a well-sorted production rollout that minimizes chances of users finding issues before the deployment staff.

- ***Redundancy***

The current number one reason for loss of service is planned and unplanned electrical outages. Our archive spans four different raised floor environments. Consequently, we often react to regional raised-floor power work for nearby projects and our own expansion. This is exacerbated by newer electrical safety rules that prohibit electricians from performing “hot work”.

To mitigate, certain hardware has been duplicated in redundant configurations.



Metadata disk for the HPSS Core Server is replicated in two different rooms 1km apart. Either can go down and the application will continue operation. Using operating system disk mirroring on top of RAID controllers across highly available duplicated fiber channel switching technology ensures no one single point of failure between the main core server and its metadata.

Further, robotic software control servers (for Oracle ACSLS) have been made redundant in a cold spare configuration, also located 1km apart.

Identifying single points of failure allows us to concentrate on the biggest bang for our redundancy dollar. The disk and tape mover nodes exist in smaller commodity hardware configurations, in sufficient quantities, so as to allow for individual node failures. Failed nodes are fenced out by our scalable application, HPSS, all while the remaining movers handle the load.

Core server hosts, on the other hand, can be found to have redundant internal drives, fiber HBAs, fans, ethernet cards, power supplies, and ECC memory.

PDUs are specified for twin tailed power sources and are fed from two panels where available.

- ***Measured doses of code patching***

Keeping up the nines of availability requires resisting the urge to over-patch the production systems. Security concerns should be thought out and patches tested cohesively in pre-production environments. With few exceptions, the most egregious software security vulnerabilities can be handled by a workaround or an efix which keeps the main archive service available without interruption. Constant patching equals constant downtime.

III. EFFICIENCY

In many ways, data archives are a study in how to do more with less. Budgets and personnel tend to not grow in step with storage requirements.

- ***Trim the fat***

With enough inexpensive data mover hosts, expensive-to-purchase and even more expensive-to-maintain fiber switch technology is not required.

Our data movers are commodity hardware based x86_64 systems running Linux. All devices are direct attached to the HBA on the host in either FC4 or FC8 native speeds. Fiber trunks running to patch panels handle the interconnects. No electrical is required to these panels.

Should one of these systems crash, there are plenty of remaining nodes to shoulder the load. We mark their associated devices unavailable to the archive application, thus no need exists for a switching architecture to swing devices to online hosts.

- ***Piggyback procurements***

Given this commodity hardware data mover design, we are able to leverage the sorts of purchases HPC centers make all the time, namely large cluster and file system disk purchases.

With modest adjustments of node configurations, what was a compute node can be a quite capable and inexpensive I/O data mover machine if tied into the larger procurement process.

- ***Vendor manpower***

Our center has dedicated operations staff well versed in the various hardware types and associated common failure scenarios. Specific vendor gear exists onsite in considerable quantities. Accordingly, we find it possible to negotiate daily onsite vendor CSE/CE support at modest rates. This allows us to have a specialist available for the inevitable unique problems falling outside the scope of an operations staff, as well as for providing a fast track to backend developer support at a moment's notice. This speeds time to resolution and frees our staff to concentrate on the administration of the archive and center at large.

- ***Authoritative sources of information***

An essential component of archive management involves reliably answering questions whose result set changes from frequently to hardly ever. Sources for such questions range from automated scripts to reports written for management. Examples include:

- What milestones were achieved last year?
- What are the firmware versions on the tape drives?
- What fixes make up our previous production code release?

Establishing a single authoritative source abates confusion. The authoritative source often differs for each question, but needs to be identified and communicated to avoid future errors based on incorrect or drifting information gathered from substandard sources (e.g. a file in team member's home directory).

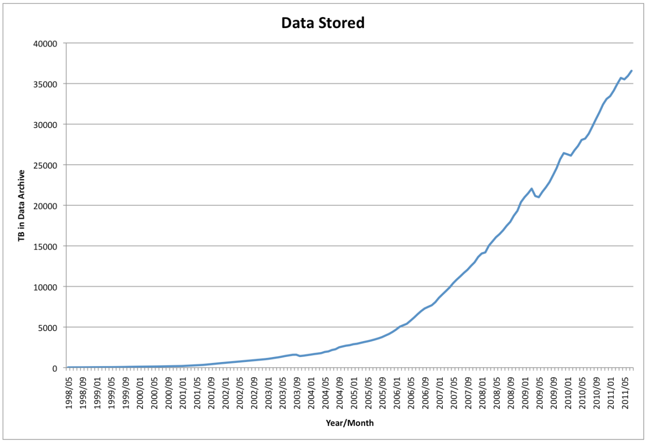
For example, the archive team coalesced on a TeamForge (SourceForge) web utility, which provides a wiki and source control among others. We track project progress here, create How-To's, load key diagram documentation, etc.

Using tape drives as an example, we write utilities that get information in real time by accessing drives over their built in Ethernet connections. Items such as dump status, firmware version, currently mounted cartridge, feet of tape processed, etc. can be gleaned in this fashion.

Our application code and the various local modifications are kept in subversion. We track preproduction and production series. The team members checkout the code, interact with it, and check it back into the central repository. All changes are logged.

MANAGING GROWTH

Fiscal year 2011 marks the first production year of our new Archival Quota system (a.k.a. Aquota). Traditionally, users have been allowed to grow our data archives with few restrictions.



Growth in the last few years suggested that we would need to construct vast new buildings to hold data if this growth curve was to be sustained.

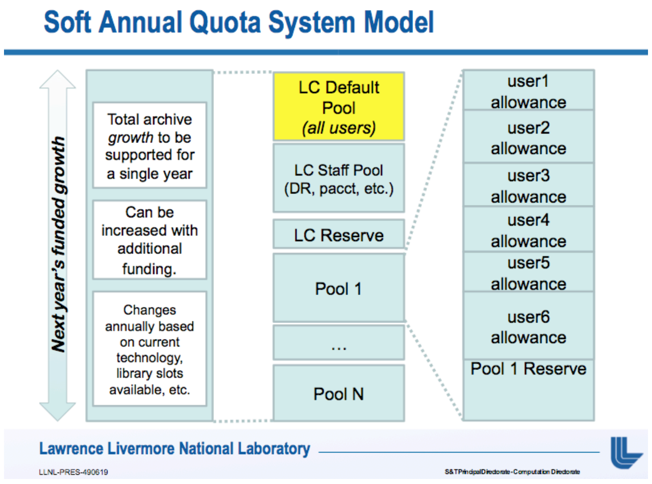
- ***Unique to this quota system***

Two key differences exist comparing Aquota and a traditional disk quota. First is that only annual growth is measured. Data stored the fiscal year prior and before is not considered. Quotas are reset each new fiscal year.

Secondly, it is "soft" enforcement only. Users are still allowed to store after their limits are reached. Users as well as their responsible program managers are contacted when quota is met. It is reported that they have grown beyond their default allowance and need to seek additional resources.

- ***Aquota Model***

Most users live within their yearly budget. The center allocates "pools" of storage to projects. Individuals exceeding their default allowance need to be given space from project pools.

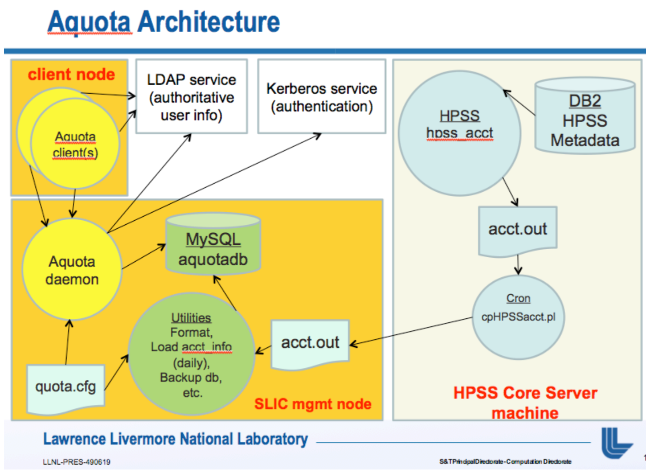

Soft Annual Quota System Model
Next year's funded growth
Total archive growth to be supported for a single year
Can be increased with additional funding.
Changes annually based on current technology, library slots available, etc.
LC Default Pool (all users)
LC Staff Pool (DR, pacct, etc.)
LC Reserve
Pool 1
...
Pool N
user1 allowance
user2 allowance
user3 allowance
user4 allowance
user5 allowance
user6 allowance
Pool 1 Reserve
Lawrence Livermore National Laboratory
LLNL-PRES-490619

This model of growth control allows the center to predetermine the amount of growth it is willing to sustain for the upcoming fiscal year, rather than attempting to budget based on the previous year's unabated growth.

Once a budget is set, a reasonable set of growth constraints is arrived at based on the amount of media the budget will allow (including potential technology refreshes).

- ***Aquota Architecture***

Aquota was built in-house. It is comprised of a server daemon written in C, any number of multiple interactive clients written in C, and a variety of administrative tools written in Perl.



Nightly exports of HPSS accounting data are imported into a MySQL database. The Aquota daemon handles all client Aquota requests, which can run on a variety of hosts in the center. Users, Pool Managers, and Administrators have increasing levels of authority and interface with the system via the command line client.

- ***Impact***

Early evidence for FY11 suggests that overall annual growth will have dropped 14 points from the previous three-year average. The tangible impact is that a tool to facilitate a dialogue has been opened between users, responsible managers, and those of us tasked with offering the archive service. This did not exist in previous years. A common language is now being spoken.

CONCLUSIONS

Data archives outlive architectures, operating systems, and interconnects. They grow with wild abandon. Bytes churn in a maelstrom of activity as new data arrives and old data is repacked.

Even with a cadre of the latest technological advances and efficient models of deployment, the primary elements of a successful data archive are the people and their willingness to strive to meet the credos of the archive. Key skills in computer science - particularly in languages interpreted and compiled - don't hurt either.

Challenges in Managing Multi-System Multi-Platform Parallel File Systems

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Ryan Braby
National Institute for Computational Sciences
rbraby@utk.edu

Rick Mohr
National Institute for Computational Sciences
rmohr@utk.edu

ABSTRACT / SUMMARY

The National Institute for Computational Sciences (NICS) is looking to deploy one or more center wide parallel file systems. Doing so should reduce time to solution for many NSF researchers. Researchers who run on multiple systems at NICS will no longer need to move data between parallel file systems and hopefully this will reduce the amount of file system space used for extraneous data replication. However, there are a number of challenges in setting up a multi-system, multi-platform parallel file system. This paper discusses many of the identified challenges for deploying such a file system at NICS and supporting at least the following architectures; Cray XT5, SGI UV, and commodity Linux clusters.

INTRODUCTION

The National Institute for Computational Sciences (NICS), a partnership between the University of Tennessee and Oak Ridge National Laboratory, was granted a $65M award from the NSF in September 2007. A series of Cray HPC systems, named Kraken, were purchased and deployed. Currently, Kraken is a Cray XT5 system with a peak performance of 1.17 PFlops. Lustre is the primary file system for Kraken, and it is built on top of DDN storage directly attached to special I/O service blades in the Cray. These blades act as the MDS and OSS servers for the rest of the system.

In the last year, NICS has deployed a new file system to be shared between Nautilus (a large SGI UV) and Keeneland (a cluster used for GPGPU development). An evaluation of file system technologies was done, and Lustre was selected for use here. Ideally, the scratch file systems will be shared across all NICS HPC resources. To this end, we have been planning and preparing to upgrade our Infiniband SAN, attach Kraken to this SAN, and migrate Kraken's current Lustre file system to be SAN attached.

While a number of sites have deployed multi-cluster Lustre file systems, unique site requirements prevent the creation of a one-size-fits-all solution. NICS supports a wide variety of platforms (Cray XT, SGI UV, and Linux clusters). Individually, these platforms can present challenges for a site-wide Lustre file system. Combining them further complicates matters.

CRAY XT

Cray ships Lustre as part of CLE (Cray Linux Environment), but they are currently shipping an older version (1.6.5) with custom patches. While it is nice to have a vendor supported version, this version is older and lacks features that have been introduced in newer versions. As we move to a center wide file system, there are also concerns about version compatibility between the servers and all the clients.

While it should be possible to put a newer version of Lustre into CLE boot images, there are a number of possible complications with doing so. At this time NICS does not have a file system developer and it is not in our short term plans to hire one. We could build and install Lustre, but we have minimal resources to test it on. Lacking a file system developer our abilities to fix issues with Lustre in CLE would be limited. The Cray XT systems use a proprietary SeaStar network, which requires it's own Lustre Network Driver (LND) and could complicate LNET routing. Further, Cray support might be hesitant to help on production issues when we are running our own version.

SGI UV

The SGI UV, is a large NUMA architecture with a single system image. Running a single Linux kernel, this architecture tends to get poor IO bandwidth when compared to clustered systems of similar core count. NICS has spent time testing multiple file systems on our 1024 core UV system, and determined that in present day performance Lustre (1.8.6) was the winner (just barely).

Comparing the known road maps for the major parallel file systems, Lustre was the only one that has plans for improving SMP scalability and NUMA performance. In particular, it looks like some improvements in this area have already been added in Lustre 2.1.

Another challenge for parallel file systems on the SGI UV is effectively utilizing multiple network interfaces to our SAN. As a large single system image system, it is important for performance that a file system can drive multiple network interfaces at near line rate. We have had some success scaling Lustre read performance with multi rail infiniband on our UV. This is an area that we hope to see improvements to Lustre for in the future.

LINUX CLUSTERS

Linux clusters with Infiniband interconnects are probably the most common platform for Lustre file systems. As such, including Linux clusters in a multi-cluster Lustre configuration adds some to the complexity. It is another platform to consider and keep track of, but it is also one that you can rely on the community for testing and development.

MULTI-SYSTEM CHALLENGES

Deploying a Lustre file system that spans multiple systems and architectures introduces new challenges apart from the previously mentioned system-specific ones. For example, it may be desirable to run Lustre 2.1 on the SGI UV in order to address some of the SMP scalability issues. However, this would require running Lustre 2.1 on the MDS and OSS servers, which is not compatible with the Lustre 1.6 client on the Cray.

Maintaining compatibility between all of the clients and servers is the first major challenge to a multi-system Lustre deployment. Different platforms may require different patches, and in some cases require different client versions. Knowing which versions are compatible and testing the compatibility is critical to ensuring file system usability.

Some system vendors include a supported version of the Lustre client and publish supported client / server combinations. Merging these requirements from multiple vendors could lead to a situation where the supported versions are not compatible with each other. To reconcile this may require running a version not supported by one or more vendors. One approach to deal with this would be to purchase third party Lustre support.

Managing a multi-system parallel file system makes the file system more of an infrastructure service. Since multiple rely on the availability of the file system, the effects of any disruptions (like maintenance) must be carefully considered. Further, you have to plan upgrades carefully; ensuring that at all points in your upgrade plan you are on compatible versions and not unintentionally running an unsupported combination of server, router, and client versions.

Also, like any infrastructure service, there are possible contention issues. Performance on one system can and will be impacted by access from another system.

CONCLUSIONS

NICS is planning to move to center wide Lustre file systems. There are a number of issues involved in doing this. While we have identified many of the issues and have ideas of how to deal with them, we do not have the experience and history of implementing these ideas to determine if they are indeed best practices.

REFERENCES

1. T. Baer, V. Hazlewood, J. Heo, R. Mohr, J. Walsh, *Large Lustre File System Experiences at NICS.* CUG 2009, http://www.cug.org/5-publications/proceedings_attendee_lists/CUG09CD/S09_Proceedings/pages/authors/11-15Wednesday/12B-Walsh/walsh-paper.pdf
2. G. Shipman, D. Dillow, S. Oral, F. Wang, *The Spider Center Wide File System; From Concept to Reality.* CUG 2010, http://www.nccs.gov/wp-content/uploads/2010/01/shipman_paper.pdf
3. Whamcloud JIRA for Lustre SMP salability: http://jira.whamcloud.com/browse/LU-56

U.S. Department of Energy Best Practices Workshop on
File Systems & Archives
San Francisco, CA
September 26-27, 2011
Position Paper

Kevin Harms
LCF/ANL
harms@alcf.anl.gov

ABSTRACT / SUMMARY

This paper addresses the "Usability of Storage Systems" and the "Administration of Storage Systems" topics. Given the small staff assigned to the LCF Intrepid storage resources we have searched for methods to optimize the use of our storage resources. We present these methods for proactively finding opportunities to tune application I/O and finding degraded hardware that is reducing overall I/O throughput.

INTRODUCTION

The LCF is a relatively new facility and is in the process of developing its storage practices and procedures. One item that has become clear is the need to be proactive about storage usage and administration. We have a limited staff dedicated to the storage system and waiting until an issue turns into a real problem leaves us in an awkward position. In order to address this, we are working on some methods to proactively find issues and start working to solve them before they become worse.

The first method was to install a tool, Darshan, to profile user I/O so that users and staff could have a basic tool to help tune I/O for Intrepid and best utilize the storage resources LCF provides.

The second method is to begin looking at the overall performance of the storage hardware to find and fix marginal hardware without the need to wait for it to degrade to the point of outright failure.

System Description

Here is an overview of the core Intrepid storage system. Intrepid has two main storage systems. The home file system is GPFS based and uses 4 DDN9550 SANs that are directly attached via DDR IB to 8 xSeries file servers. The scratch storage has two different file systems running on it, GPFS (intrepid-fs0) and PVFS, (intrepid-fs1) which utilize the same hardware. The scratch area uses 16 DDN9900 SANs, which are directly attached via DDR IB to 128 xSeries file servers. (8 servers per DDN) File system clients are connected over a 10 GB Myrinet fabric.

THE USABILITY OF STORAGE SYSTEMS

Darshan

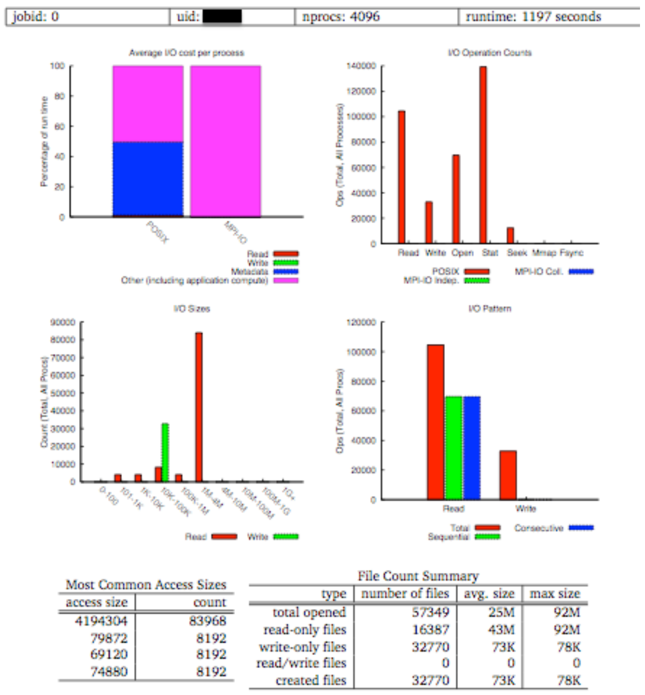
Darshan [1] was a tool developed by the MCS department in ANL and deployed on the LCF Intrepid Blue Gene machine. Darshan captures information about each file opened by an application. Rather than trace all operational parameters, however, Darshan captures key characteristics that can be processed and stored in a compact format. Darshan instruments POSIX, MPI-IO, Parallel netCDF, and HDF5 functions in order to collect a variety of information. Examples include access patterns, access sizes, time spent performing I/O operations, operation counters, alignment, and datatype usage. Note that Darshan performs explicit capture of all I/O functions rather than periodic sampling in order to ensure that all data is accounted for.

The data that Darshan collects is recorded in a bounded (approximately 2 MiB maximum) amount of memory on each MPI process. If this memory is exhausted, then Darshan falls back to recording coarser-grained information, but we have yet to observe this corner case in practice. Darshan performs no communication or I/O while the job is executing. This is an important design decision because it ensures that Darshan introduces no additional communication synchronization or I/O delays that would perturb application performance or limit scalability. Darshan delays all communication and I/O activity until the job is shutting down. At that time Darshan performs three steps. First it identifies files that were shared across processes and reduces the data for those files into an aggregate record using scalable MPI collective operations. Each process then compresses the remaining data in parallel using Zlib. The compressed data is written in parallel to a single binary data file.

Darshan was deployed on Intrepid by creating a modified set of mpiXXX compiler wrappers which link in the darshan library code. These modified compiler wrappers are part of the users default path which means many applications link in Darshan with no extra work by the user. These applications put logfiles into a common area and are setup so only the user who produced the logs can read them. Later we change the group permission to a special 'darshan' group and then add group read permission. These logs then become accessible by the LCF staff and a few selected MCS research staff.

User Analysis

The first capability this provides is for users to look at some information about their jobs I/O profile and compare it to common suggestions available via our wiki documentation. If the user feels their I/O performance is not as good as it should be, when contacting the LCF staff, we already have some basic information about the I/O patterns they are using which might give some initial starting suggestions for the user to try for improving I/O performance on Intrepid. This also addresses a common issue where users are not familiar with how their application does I/O, perhaps because they are using some large application where someone other person or group implemented the IO code. Figure 1 shows an example from the *darshan-job-summary.pl* output.



jobid: 0	uid: [redacted]	nprocs: 4096	runtime: 1197 seconds

Most Common Access Sizes

access size	count
4194304	83968
79872	8192
69120	8192
74880	8192

File Count Summary

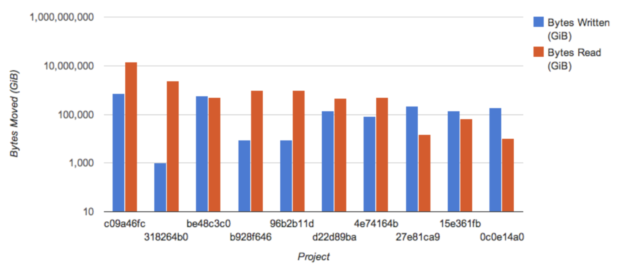
type	number of files	avg. size	max size
total opened	57349	25M	92M
read-only files	16387	43M	92M
write-only files	32770	73K	78K
read/write files	0	0	0
created files	32770	73K	78K

Figure 1 – Darshan Job Summary Example

This summary information can provide a useful starting point for I/O analysis. We are aware of a few applications that have used this output to successfully improve their application I/O for Intrepid.

Project Analysis

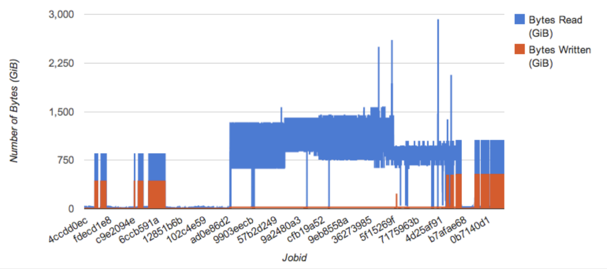
The second capability is to proactively analyze darshan logs to see how users are utilizing the storage system and if they are being effective. We are developing a basic web interface around aggregated log files that can be examined on a per-project basis to find who the major users of the storage system are and how are they using the system. We explored this idea in reference [2]. Figure 2 shows the top 10 projects from 1/1/2011 to 6/30/2011. We can look at these projects individually to see how they are using the I/O system.



Project	Major Project	Bytes Written (GiB)	Bytes Read (GiB)	Bytes Moved (GiB)
c09a46fc	incite	717960.320269	14378691.7527	15096652.073
318264b0	incite	1028.50599501	2415161.5417	2416190.0477
be48c3c0	incite	583163.035626	500354.756459	1083517.79209
b928f646	alcc	8999.55220313	970914.933426	979914.485629
96b2b11d	incite	8912.51224884	940718.474662	949630.986911
d22d89ba	incite	141647.703103	447793.368016	589441.071119
4e74164b	incite	81074.8510083	489839.286861	570914.137869
27e81ca9	incite	222602.486601	14497.4064404	237099.893041
15e361fb	incite	144678.540998	64092.520676	208771.061674
0c0e14a0	incite	181726.374965	10113.3177587	191839.692723

Figure 2 – Top 10 Projects by bytes moved

Once we identify the top I/O users we can examine their I/O usage in more detail. Figure 3 shows an example of aggregate information about a single project. We can look at the percent time spent in I/O and see if we should consider talking with a project about their I/O usage if it looks subpar and thereby improve their utilization of the core-hours they have been granted.



Project	Major Project	Bytes Written (GiB)	Bytes Read (GiB)	Bytes Moved (GiB)
c09a46fc	incite	822788.152176	14693539.2635	15516327.4157

Project	Min (MiB/s)	First Quartile (MiB/s)	Second Quartile (MiB/s)	Max (MiB/s)
c09a46fc	0	42.270659	4827.955307	11199.311494

Project	Job Coverage	Job Coverage Ratio
c09a46fc	1799/3315	0.542684766214

Project	Percent Time in I/O
c09a46fc	0.289274515813

Figure 3 – Darshan Project Information

We had identified a project in 2010 that was a top I/O consumer but had a high percentage of time spent in I/O. We were successful in working with this project to change the method for writing of files, which gave them a 30% improvement in write throughput.

System Planning

The third capability we get is the ability to look at what I/O patterns users want to use and what they want to do. This information can be used to target how we allocate our resources for next generation systems. Examples from above show users are still obsessed with generating O(1000), O(100000), O(1000000) files. The file per process model tends to break down at the 8192 node level (or 32768 processes) on Intrepid. For our next generation system, we have planned to split data and metadata and use separate SSD based storage for the metadata in hopes of boosting metadata performance, which would serve as a band-aid for the file-per-process users.

Another point is that we see about 60% of the jobs at large scale go to either shared or partially shared files and fewer use the file-per-process model. Figure 4 shows this distribution. However, in this same time period we saw remarkably few people using high-level libraries such as HDF5 or PnetCDF. This might indicate we need to spend time educating the userbase about these libraries or find out why our users would rather create their own shared file format rather than leverage existing ones.

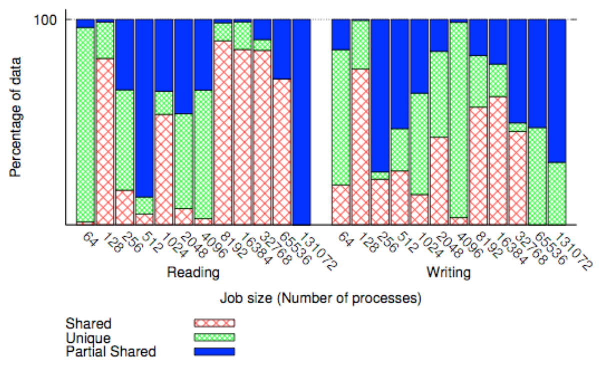


Figure 4 – File usage (1/2010 – 3/2010)

THE ADMINISTRATION OF STORAGE SYSTEMS

Another aspect of storage system efficiency is ensuring the current hardware is delivering performance up to its useable peak. Anecdotally we have observed a single failing SATA disk can produce a global slowdown of the scratch file system. During our early test stages when we were tuning the /intrepid-fs1 file system, we would often find a marginal drive would cause a significant slow down in an IOR test case. As an example, we would see something on the order of losing 50% of total throughput. After failing 1 (or more) drives, the system would return to its optimal performance level.

The work we have done in this area is still very preliminary and we have not validated any of our suppositions.

Log Analysis

The DDN9900 will report many errors and statistics but it also logs informational events in the system log. These are generally not reflected directly in any of the system statistics. We developed a trivial monitoring tool to check the event logs of each DDN approximately once per day and send and alert if there were a large number of new messages in the log. Here is a short snippet from the monitoring tool, which emails its results.

```
        INFO  INT_GH    8-29  12:50:31  Recovered:
Unit Attention Disk 9G GTF000PAH51JNF

        INFO  INT_GH    8-29  12:59:29  Recovered:
Unit Attention Disk 22G GTF002PAHHKXRF

        INFO  INT_GH    8-29  13:02:43  Recovered:
Subordinate errors detected.

        INFO  INT_GH    8-29  13:05:18  Recovered:
Unit Attention Disk 2G GTF100PAHW59BF

        INFO  INT_GH    8-29  12:49:44  Recovered:
Unit Attention Disk 13G GTF002PAHWD21F

        INFO  INT_GH    8-29  13:00:31  Recovered:
Subordinate errors detected.

        New Log Messages: 2650
```

Example 1 – DDN Log monitoring output

In Example 1, we see that this DDN had 2600 new log messages and many of those messages are related to problems with disk access on channel ‘G’. In this case, we could have opened a support request with DDN to determine which component was really at fault. In this particular case, disk 7G failed 5 days later. We could have failed disk 7G earlier and presumably not lost any performance during that time period.

Visualization

Another method to monitor the storage infrastructure for marginal components is via visualization of I/O metrics. We setup a utility to pull the ‘tierdelay’ metric from all tiers of each of the 32 DDN controllers associated with the scratch file system. We then ran the IOR benchmark with a write workload while we collected samples every 10 seconds. The data was loaded into ParaView and we began looking for patterns.

Figure 5 shows a combined visualization of total operation count for each channel/tier combination for all DDNs at the last timestep.

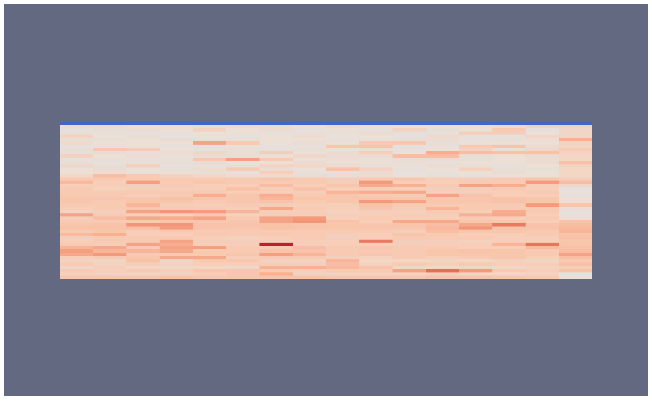


Figure 5 – Cumulative Operations

Since the IOR workload was evenly distributing data over all LUNs we should see similar operation counts, but instead we see one tier (dark red) that has significantly more operations than the rest of the tiers. In talking with DDN, the ‘tierdelay’ counter records all operations including internal retries. It would appear that there is some issue on this particular tier resulting in retries being generated.

Figure 6 shows the same metric again but now as a 3D volume.

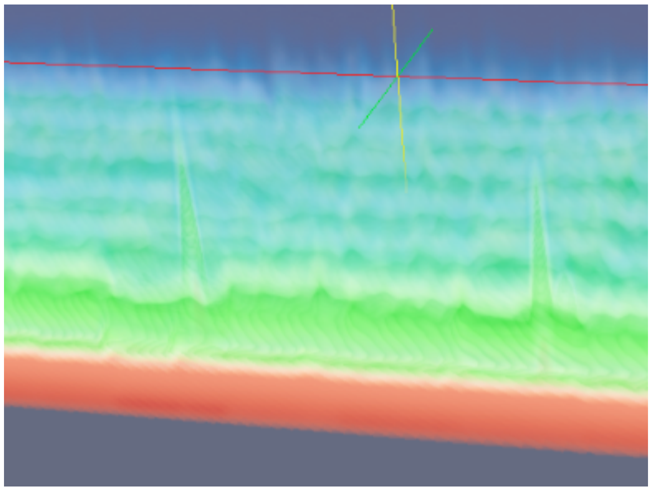


Figure 6 – Tier Delay as Volume

The volume shows a count for the number of operations, which occurred within a defined bucket. For example, 100 operations at 0.2 second delay. The bottom of the volume is the shortest delay and top of the volume is the highest delay. The dark red coloring are higher counts going to blue at the lowest counts. In general the image shows the lowest latency buckets have the highest counts, which is good and the highest latency buckets have the lowest counts, also good. However, you can see a spike on a couple of disks where the higher latency buckets have a much higher total count than most other disks. We don't have conclusive findings that those disk are causing system wide problems, but that is an example of what we hope to find.

CONCLUSIONS

The Darshan deployment has been successful on the LCF Intrepid system. A few projects have used it to tune I/O characteristics to optimize for Intrepid and seen improvements in throughput. We also identified a project that was significant storage user but also suffered from slow I/O performance. We worked with the members of this project to update their code with a slightly modified I/O model that used fewer files which resulted in a 30% improvement of their I/O write speed. We plan to continue to enhance our summarization web tools to provide easier access to the darshan data for the LCF staff.

Our progress on identifying faulty hardware prior to failure on the DDN SANs is still very preliminary and we have not validated any of the results. We hope to progress this further by being able to validate performance improvement after a hardware replacement. We would also hope to identify these patterns so that we could create statistical models that would work on the normal I/O load of Intrepid without the need for an invasive diagnostic run.

ACKNOWLEDGEMENTS

Phil Carns – for all that is Darshan

Neal Conrad – for Darshan web development

Justin Binns – for IO visualizations

REFERENCES

1. Philip Carns, Robert Latham, Robert Ross, Kamil Iskra, Samuel Lang, and Katherine Riley. 24/7 characterization of petascale I/O workloads. In *Proceedings of 2009 Workshop on Interfaces and Architectures for Scientific Data Storage*, September 2009.
2. Philip Carns, Kevin Harms, William Allcock, Charles Bacon, Samuel Lang, Robert Latham, and Robert Ross. Understanding and improving computational science storage access through continuous characterization. In *Proceedings of 27th IEEE Conference on Mass Storage Systems and Technologies (MSST 2011)*, 2011.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives
San Francisco, Ca
September 26-27, 2011
Position Paper

Roger Combs
Navy LSRS Program
roger.combs1@navy.mil

Mike Farias
Sabre Systems
mfarias@sabresystems.com

ABSTRACT
The Navy Littoral Surveillance Radar System (LSRS) Program has demanding streaming aggregate I/O requirements (double-digit GB/sec level). The LSRS Program also has petabyte-level data management issues and accompanying data management policies and procedures that are under constant review.

INTRODUCTION
This document will address current Navy LSRS best-practices within our own High Performance Computing (HPC), capacity environment. Areas of concern will be the following:

a. Business of storage systems
b. Administration of storage systems
c. Reliability of storage systems
d. Usability of storage systems

Business of storage systems: Currently the LSRS Program uses Oracle Storage Archive Manager/Quick File System (SAM/QFS) as the parallel file system and respective Hierarchical Storage Management (HSM) solution to meet our data storage and management needs. Strategically, business viability of SAM/QFS under Oracle, post-Sun Microsystems acquisition, has and continues to be a major concern. As a result of several meetings with Oracle concerning SAM/QFS, ultimately the IBM General Parallel File System (GPFS) and the High Performance Storage System (HPSS) were chosen as the future file system/HSM solution. From both production experience and consensus among some DoE colleagues, a parallel file system is currently regarded as the most challenging and critical aspect of HPC operations, frequently referred by LSRS as the "backbone." As fallout of this "backbone" ideology, when faced with an acquisition decision regarding SAM/QFS, only two file systems came into play. Criterions for selection were items such as company viability, development talent, and a deep R&D budget / bench. Ultimately, this list revolved around two solutions, Lustre/HPSS and GPFS/HPSS. Cost was not a criterion for parallel file system selection for CY12 migration.

Historically, cost was a criterion for selection of our SAM/QFS file system and our current migration efforts are serving as a lesson-learned. Moving forward, there has been concern about the viability of the SAM/QFS parallel file system beyond CY11 in terms of development and support. For our "backbone," there also have been concerns with Lustre and Oracle IP strategy potentially being an issue. Concerns with Lustre stability were also negatively factored into the decision process from reading publications such as the Livermore National Lab (LLNL) I/O "Blueprint" from 2007[1].

From a business perspective, LSRS best practices dictate that the "backbone" be the most performant solution that can be afforded under the company with the deepest R&D bench. An additional requirement is that the provider of the parallel file system middleware be relevant in the HPC marketplace. Storage acquisition (both cache and tape) are approached from a best-of-breed perspective and not a cost perspective.

Administration of storage systems: Currently, storage system administration is handled and led entirely by private industry personnel. Strategically, LSRS recognizes that this is not best practices, and future administration and management of storage systems will have a government technical lead. Across all HPC functional areas, there will be government division leaders aka department heads (DHs).

Above and beyond organizational layout, monitoring and benchmarking tools could always be better for storage infrastructure in general. Interleaved or Random (IOR) benchmarking is used to get theoretical maximums for I/O on capacity storage. Above and beyond, IOR, solutions from companies such as Virtual Instruments have been explored to potentially better capture Fibre Channel I/O in near-real-time and identify bottlenecks. However, currently Virtual Instruments does not support or project to support Quad Data Rate (QDR) Infiniband, which is orthogonal to our HPC I/O roadmap.

In general, parallel I/O benchmarks seem limited and a bit immature given the projected requirements for data-driven computing currently and in the future[2]. From a tape perspective, minimizing media that is more than a generation behind the current industry products is policy. While tape certainly has value, from our production experience, lifecycle management of tape has proven to be challenging. Subsequently, we are facing the task of ascertaining if obsolete media needs to be discarded or go through a relatively painful conversion process.

Reliability of storage systems: Organizationally, file system reliability is believed to be directly related to file system scalability and stability. From this, we borrow from the 2007 LLNL I/O Blueprint[1], in asserting that in general, file systems are sized to no less than three orders of magnitude below the compute platform(s) they support, i.e., a 10TF system would need no less than 10GB/sec of aggregate I/O bandwidth behind it. Leveraging this approach has significantly increased productivity and nearly eliminated staging. In support of consistent systems reliability and balance, file system and network interconnect acquisition precedes platform acquisition. Systems acquisition is also approached from a modular perspective in similar fashion to Mark Seager's Peloton and associated Hyperion based initiatives at LLNL.

By definition, we assert that file systems that are not horizontally scalable are intrinsically unstable. QFS currently suffers from the preceding quality with one metadata server per namespace. The current, QFS file system is monolithic; LSRS has established a requirement for no less than two production (primary and secondary) parallel file system namespaces for capacity high-availability. As disk caches for HPC centers enter the petabyte and beyond level, we've found from production that file system scalability capabilities do not necessarily hold up to vendor claims. To provide continuity of daily operations, it is critical that two namespaces are on the floor ready-to-go at any given time. From experience, edge-cases are frequently encountered, taking days or more oftentimes weeks to solve. The preceding service-losses or impacts are compounded when cache-repopulation is considered with file systems at the petabyte level taking weeks to re-populate. With QFS particularly, in terms of monolithic metadata architecture, and the associated production issues that resulted, LSRS realizes the importance of choosing superior architectures and support organizations. LSRS metadata storage is handled from an IOPS-centric point-of-view and RAMSAN technology is used for metadata storage. As a backup, physical solid-state disk is used to complement the RAMSAN. While one monolithic namespace has performance advantages, we plan to leverage two namespaces in the very near future. Post QFS-migration, two GPFS namespaces will be established, prior to QFS-migration a single QFS and GPFS (Data Direct Networks SFA10KE "Gridscaler") namespaces will be established.

Furthermore, to manage job quality of service, Navy LSRS borrowed from the DoD High Performance Computing Modernization Office (HPCMO), and established their Normalized Expansion Factor (NEF) Metric[3]. The details of this metric can be found in an FY2002 whitepaper from HPCMO referenced below, but essentially the metric is a normalized way to measure job quality with no queue-wait time at-all associated with jobs having a NEF of 1.0.

Heuristically, high priority work should not exceed an NEF of 1.7, whereas standard workload should not exceed 2.2. NEF metrics are collected for each individual job and performance data is kept indefinitely.

Usability of storage systems: To address usability, LSRS strategically attempts to minimize the number of namespaces deployed to two vice, having multiple in the past. The preceding has obvious usability advantages, but also the performance advantage of having more drive spindles under one namespace. Block-level storage, in general, is abstracted away from analysts using in-house developed mass-storage APIs. In our environment, usability is dominated by performance and concessions are viewed as necessary. Generally, performance, scalability, and stability are the three dominant factors in strategic file system thought. Usability is still a distant fourth-level consideration.

CONCLUSIONS

The most important aspects of file system and archive best practices are an understanding that the system design-points need to be a function of the application sets, both current and projected, running in production. Heuristics will get you close to balanced, and generally keep architects out of trouble, but to really get outstanding performance requires closer interaction with analysts. At Navy LSRS, the file system is currently regarded as the "backbone" of production operations and subsequently a lot of attention is paid to ensuring that it is sized properly and has an appropriate interconnect and bandwidth.

Tape is effectively sized using the write rate of a typical run of the most write-intensive application in production. While certainly valuable and viable for the long-term, tape has presented Navy LSRS with a number data lifecycle management issues regarding a myriad of end of life tapes and infrastructure (silos). Many of these tapes have questionable value, but due to this uncertainty, they create a lot of work in mining data from useful media and discarding useless media. While valuable, tape certainly presents maintainability issues if allowed to veer too far from current generations and formats.

Additionally, from a business perspective, much has been learned in terms of interacting with vendors as well as integrators and reading between the lines. From an organizational perspective, Navy LSRS has shifted into an organization that is much more critical of consumed information than in the past. The preceding applies across all functional areas. In other words, asking "is what the vendor is saying useful," or "is what our integrator is saying practical?" All too often, initially, answers were frequently no. Oftentimes, further investigation led to invaluable insights into real vendor positions vice stated, or performance improvements that were never realized due to inadequate architectural and or operations planning. Particularly with file system and archive materiel, betting on the wrong technology or vendor can be costly, well into the seven-figures and beyond. Subsequently, staying in tune with the HPC community has proven to be a very fruitful investment of both time and energy.

Finally, establishment of file system and I/O roadmaps, i.e., LLNL's I/O "Blueprints" has helped Navy LSRS tremendously. Moving from ad hoc approaches to file system and archive operations to planned and deliberate signed documentation has forced our organization into making much more informed decisions. Roadmaps, in general are key in supporting a successful HPC program.

REFERENCES

1. Wiltzius, D. et al. (2007). *LLNL FY07 I/O Blueprint*, 2007. Retrieved from LLNL website: https://e-reports-ext.llnl.gov/pdf/342161.pdf
2. Cook, D. et al. (2009). *HPSS in the Extreme Scale Era*, 2009. Retrieved from NERSC website: http://www.nersc.gov/assets/HPC-Requirements-for-Science/HPSSExtremeScaleFINALpublic.pdf
3. DoD High Performance Computing Modernization Office (HPCMO) (2002). *HPC System Performance Metrics*, 2002. Retrieved from HPCMO website: http://www.hpcmo.hpc.mil/Htdocs/HPCMETRIC/fy2002_hpcmp_metrics.pdf

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Oak Ridge Leadership Computing Facility Position Paper

Sarp Oral, Jason Hill, Kevin Thach, Norbert Podhorszki, Scott Klasky, James Rogers, Galen Shipman
Oak Ridge Leadership Computing Facility, Oak Ridge National Laboratory
{oralhs, hilljj, kthach, pnorbert, klasky, jrogers, gshipman}@ornl.gov

ABSTRACT / SUMMARY

This paper discusses the business, administration, reliability, and usability aspects of storage systems at the Oak Ridge Leadership Computing Facility (OLCF). The OLCF has developed key competencies in architecting and administration of large-scale Lustre deployments as well as HPSS archival systems. Additionally as these systems are architected, deployed, and expanded over time reliability and availability factors are a primary driver. This paper focuses on the implementation of the "Spider" parallel Lustre file system as well as the implementation of the HPSS archive at the OLCF.

INTRODUCTION

Oak Ridge National Laboratory's Leadership Computing Facility (OLCF) continues to deliver the most powerful resources in the U.S. for open science[*]. At 2.33 petaflops peak performance, the Cray XT5 Jaguar delivered more than 1.5 billion core hours in calendar year (CY) 2010 to researchers around the world for computational simulations relevant to national and energy security; advancing the frontiers of knowledge in physical sciences and areas of biological, medical, environmental, and computer sciences; and providing world-class research facilities for the nation's science enterprise.

The OLCF is actively involved in several storage-related pursuits including media refresh, data retention policies, and file system/archive performance. As storage, network, and computing technologies continue to change; the OLCF is evolving to take advantage of new equipment that is both more capable and more cost-effective. A center-wide file system (Spider) [1] is providing the required high-performance scratch space for all OLCF computing platforms, including Jaguar. At its peak, Spider was serving more than 26,000 clients and providing 240 GB/s aggregate I/O throughput and 10 PB formatted capacity. For archival storage OLCF uses the high-performance tape archive (HPSS). Currently HPSS version 7.3.2 at OLCF is housing more than 20 PB of data with an ingest rate of between 20–40 TB every day. This paper presents the lessons learned from design, deployment, and operations of Spider and HPSS and future plans for storage and archival system deployments at the OLCF.

THE BUSINESS OF STORAGE SYSTEMS

Storage requirements for both Spider and HPSS continue to grow at high rates. In September 2010, two new Lustre file systems were added to the existing center-wide file system. These two file systems increased the amount of available disk space from 5 to 10 PB and will help improve overall availability as scheduled maintenance can be performed on each file system individually. The addition of these file systems provided a 300% increase in aggregate metadata performance and a 200% increase in aggregate bandwidth. Additional monitoring improvements for the health and performance of the file systems have also been made.

In August 2010, a software upgrade to version 7.3.2 on the HPSS archive was completed, and staff members began evaluating the next generation of tape hardware. Implementation of this release has resulted in performance improvements in the following areas.

- *Handling small files*. For most systems it is easier and more efficient to transfer and store big files; these modifications made improvements in this area for owners of smaller files. This has been a big gain for the OLCF because of the great number of small files stored by our users.

[*] *The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DE-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes. This research used resources of the Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.*

- *Tape aggregation*. The system is now able to aggregate hundreds of small files to save time when writing to tape. This has been a tremendous gain for the OLCF.
- *Multiple streams or queues (class-of-service changes)*. This has enabled the system to process multiple files concurrently and, hence, much faster, another huge time saver for the OLCF and its users.
- *Dynamic drive configuration*. Configurations for tape and disk devices may now be added and deleted without taking a system down, giving the OLCF tremendously increased flexibility in fielding new equipment, retiring old equipment, and responding to drive failures without affecting user access.

Following this upgrade, in April 2011, twenty STK/Oracle T10KC tape drives were integrated into the HPSS production environment. This additional hardware is proving to be very valuable to the data archive in two distinct ways. The new drives provide both a 2× read/write performance improvement over the previous model hardware and a 5x increase in the amount of data that can be stored on an individual tape cartridge. Along with improved read/write times to/from these new drives, the OLCF now benefits from being able to store 5 TB on each individual tape cartridge– effectively extending the useful life of the existing tape libraries. This has allowed the OLCF to postpone its next library purchase until the first half of FY12.

The OLCF HPSS archive has experienced substantial growth over the past decade (Figure 1). The HPSS archive currently houses more than 20 PB of data, up from 12 PB a year ago. The archive is currently growing at a rate of approximately 1PB every 6 weeks, and that rate has doubled on average every year for the past several years.

Planning around such extreme growth rates, from both a physical resource perspective and an administrative perspective, while operating within a limited budget capacity, presents several challenges. The fact that tape technology and performance traditionally lags behind that of its disk/compute counterparts presents a fiscal challenge in supporting such a large delta in the amount of data taken into the archive each year. We are forced to purchase additional hardware (tape libraries, tape drives, data movers, switches, etc.) each year in order to meet operational and performance requirements. Add in the fact that much, if not the majority of the archived data needs to remain archived for multiple generations of media (a very significant amount of resources are spent in the process of repacking data from older tapes onto newer media), and a tremendous amount of money is spent simply maintaining "status quo" of the archive each year.

The OLCF recognizes that such a model of exponential archive growth is unsustainable over the long-term. We have taken steps to mitigate this problem and slow the growth rate down by introducing quotas on the amount of data users can store in their respective home and/or project areas within the archive. In addition, we recently made a request to the Top 10 users of the archive to purge any unnecessary data from the archive, and that request to voluntarily remove data yielded well over 1 PB of data that was purged from the archive.

The OLCF has two Sun/STK 9310 automated cartridge systems (ACS) and four Sun/Oracle SL8500 Modular Library Systems. The 9310s have reached the manufacturer end-of-life (EOL) and are being prepared for retirement. Each SL8500 holds up to 10,000 cartridges, and there are plans to add a fifth SL8500 tape library in 2012, bringing the total storage capacity up to 50,000 cartridges. The current SL8500 libraries house a total of 16 T10K-A tape drives (500 GB cartridges, uncompressed), 60 T10K-B tape drives (1 TB cartridges, uncompressed), and 20 T10K-C tape drives (5 TB cartridges, uncompressed). The tape drives can achieve throughput rates of 120–160 MB/s for the T10KA/Bs and up to 240 MB/s for the T10K-Cs.

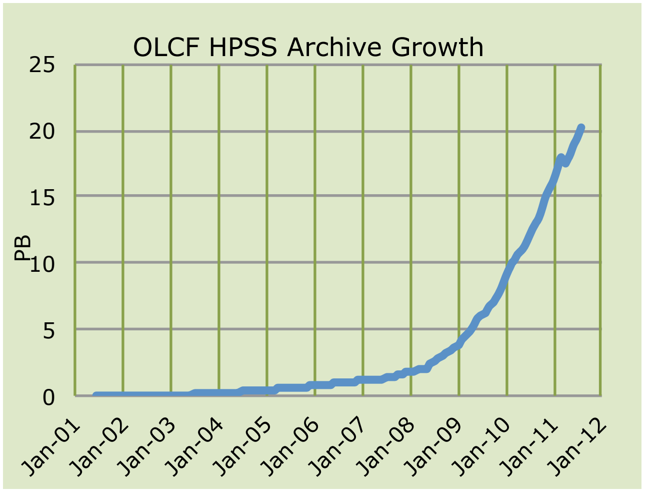


Figure 1. *OLCF HPSS Archive Growth*

OLCF follows a collaborative open source development model for its scratch space storage system. A multi-national and multi-institutional collaboration, OpenSFS [2] was formed in 2010 by ORNL, LLNL, Cray, and Data Direct Networks. The goals of the OpenSFS organization are to provide a forum for collaboration among entities deploying file systems on leading edge high performance computing (HPC) systems, to communicate future requirements to the Lustre file system developers, and to support a release of the Lustre file system designed to meet these goals. OpenSFS provides a collaborative environment in which requirements can be aggregated, distilled, and prioritized, and development activities can be coordinated and focused to meet the needs of the broader Lustre community. This collaborative open source development model allows the OLCF to have more control and input in high-performance scalable file system development. OpenSFS recently awarded a development contract for future feature development required to meet the requirements of our next-generation systems. OpenSFS has been extremely successful in organizing the Lustre community, providing a forum for collaborative development, and embarking on development of next-generation features to continue the progression of the Lustre roadmap. OpenSFS is working closely with its European counterpart (EOFS) and has signed a memorandum of understanding to align our respective communities. At LUG 2011 all communities aligned with OpenSFS providing a unified platform from which to carry Lustre well into the future, meeting not only our current petascale requirements but providing an evolutionary path to meeting our Exascale requirements.

For archival storage systems OLCF is participating in a collaborative proprietary closed source model led by IBM. OLCF is currently providing more than 2 FTEs for this collaboration. While this model provides faster development cycles and better maintenance support compared to the open source collaborative model, the cost and business related risks associated with the private company leading the development project are the drawbacks of this model.

OLCF resources are classified as medium-confidentiality and low-availability according to the Federal Information Processing Standards (FIPS). While OLCF recognizes the cost benefit of using commodity storage hardware, the current state of technology does not allow us to deploy such technology in our archival storage systems. However, as these technologies continue to mature, it might be possible to take advantage of commodity storage hardware.

THE ADMINISTRATION OF STORAGE SYSTEMS

The day-to-day administration of a large parallel file system requires coordination between not only the members of the team working on the file system (both hardware and software), but coordination throughout the computational center as these activities have the potential to cause service outages and impact performance. The OLCF has successfully deployed packages for version control of key administration scripts, as well as centralized configuration management to handle individual node configuration convergence.

The OLCF uses Nagios [3] to monitor the health of the components of the system. Custom checks have been implemented to additionally validate the correctness of the file system – specifically are the devices mounted where they are supposed to be. Additional performance monitoring of the Lustre Network layer (LNET) are done for the Lustre servers and routers in Nagios. Currently this information is not archived for future analysis it is only used for failure detection.

We use the Lustre Monitoring Toolkit [4] developed at Lawrence Livermore National Laboratory (LLNL) to grab periodic Lustre level performance snapshots. Our current dataset is quite small so it is not useful for future predictions, but we have seen interesting trends.

Finally the OLCF has written a tool that can query the Application Programming Interface (API) to the DDN S2A9900 storage controllers. We use it to monitor the performance of the backend controllers. Currently we capture Read and Write Bandwidth and IOPS. This quick glance of the overall system performance can give administrators a fast track to problem diagnosis if say the IOPS are orders of magnitude higher than the Bandwidth. In that case we know to search out an application that is using one of the Lustre OST's served by that DDN 9900.

Being a center-wide file system, Spider is key to simulation, analysis, and even some visualization for the OLCF. Great care is taken to preserve system uptime, and maintenance activities are deferred to at a minimum once per quarter downtime. This outage affects all users of OLCF compute resources, but can help to address performance issues and overall system maintenance tasks that are harder to do real-time. Much of our administrative tasks are coordinated and done live, but with the Jaguar XT5 resource in maintenance period to limit the potential issues for users if something were to go wrong. An example is the DDN controller firmware. We can upgrade one controller out of every couplet, reboot it, and then do the partner controller without causing a file system outage. This can help push the potential quarterly outage to twice per year or even once per year depending on the software releases from DDN.

After a hardware failure caused partial file loss from the Lustre file system, a full root cause analysis led to procedural changes as well as changes in e2fsprogs packages, and spurred development of fast metadata and Lustre object storage targets for determining files that are affected by a large failure of the RAID devices on the backend.

Change Control

The OLCF has used the configuration management package CFengine [5] for several years. In our implementation of CFengine we have chosen to manage configuration files at a node level (host), a cluster level (groups of hosts related by system task), the operating system level (for each version of the OS), and a generic level that applies to all systems within the center. Additional work has gone in to configure systems that are diskless requiring some workarounds within Cfengine and the rest of the OLCF infrastructure.

In our case we use it to manage the configuration of the Lustre OSS/MDS/MGS servers – we are unable to use it to manage the storage controllers themselves. Additionally we use version control to manage the configuration of the Ethernet switches and routers for simple rollback. Managing the configuration of the Lustre file system is somewhat more difficult, but we wrote scripts and configuration files that describe the file system and can be used to start/stop the file system as well as monitor the health and status of the file systems.

We can additionally use the DDN API tool to query the configuration of the storage controllers and note a deviation from the baseline configuration specified. Work is ongoing to both correctly define the baseline as well as what acceptable deviations and periods of deviation are before notifying administrators. The storage controllers are configured to send their log data to a centralized syslog server that is running the Simple Event Correlator [10] and SEC can alert for matches to pre-configured rules. We also have SEC configured to send all log data captured over a 1 hour period to the administrators for help in solving issues like failed disks or diagnosing performance problems like SCSI commands timing out.

Our current configuration management solution does not perform validation of the configuration or syntax checking for the configuration itself. The next generation configuration management solution (BCFG2) contains input validation and syntax checking on commit.

The OLCF has both development testbeds and a pre-production testbed for verifying both changes as well as system upgrades. We have however not found any bugs at this small scale that have saved problems when the change/upgrade is deployed. Some problems only reveal themselves under sufficient load.

Cyber Security

For the Spider parallel file systems at the OLCF we commit to quarterly OS patching (matching the above mentioned quarterly planned system outages), based on analysis of risk and the location in the network. This is a delicate balance of keeping the system stable/available and satisfying the desires of Cyber Security personnel in keeping systems at the most recent patch levels. The HPSS side has weekly maintenance windows (not always taken), and has the ability to roll out security patches through those windows. Outages of the archive do not affect the production compute clusters where outages for the Spider file systems would take down the compute resources.

One example of how we can demonstrate certain file systems only being available to certain nodes is via the /proc file system on the Lustre OSS and MDS servers. We have a category 3 sensitivity file system and are working to monitor the mounts of that file system via the proc file system on the Lustre servers. If a client that is not authorized to mount the file system is detected an alert is sent to the security team and logs from the non-authorized node are gathered to see who was logged in at the time of the un-authorized access.

The OLCF has three categories of data protection that map to "publicly available information" (Category 1); data that is proprietary, sensitive, or has an export control (Category 2); or data that has additional controls required based on the sensitivity, the content being proprietary, or export control (Category 3). The OLCF has a very small amount of Category 3 data and has a separate file system for Category 3 processing. The OLCF uses Discretionary Access Controls (the Unix group memberships) for controlling access to data. The OLCF project ID is a logical container for access control; where sets of users are members of projects and have access to the same information. These mappings also carry over to the HPSS archive.

Additionally the OLCF sets secure defaults for permissions on scratch, project, and HPSS directories. The default of project team only for project areas, and user only for user scratch areas, Global home areas, and HPSS "home areas". These permissions are enforced through our configuration management process (Cfengine), and users can change them by requesting the change via our help@nccs.gov e-mail address.

Managing the Unix group memberships for users closely is a requirement in our environment as these group memberships control access to data that can be considered under export controls or confidential under industrial partnership agreements. Ongoing audits of the memberships of groups is part of the day-to-day accounts processing done by our User Assistance Group and the HPC Operations Infrastructure team. Additional logging infrastructure is being setup in conjunction with the Lustre purging process developed through cooperation with Operations staff at NERSC.

Technology refresh

A key goal of the Spider parallel file system was to decouple the procurement and deployment of storage systems with that of large-scale compute resources at the OLCF. This goal has been realized and we are currently in the process of procuring our next generation file system for OLCF. To ease transition to these new file systems, for a period of 1 – 2 years, the current generation and next generation file systems will be operated in parallel. We have had success in migrating users between file systems, but the process is not without pitfalls and prone to compute users not paying attention to e-mail notifications and then having their jobs terminate abnormally as their application may expect to use a file system that is no longer in operations. Operating the file systems concurrently will allow users to make a gradual transition thereby minimizing the impact to our users.

Based on our enhanced understanding of I/O workloads of scientific applications, garnered from over 12 months of continuous monitoring of our file system environment coupled with a detailed understanding of our applications I/O kernels, we have developed an extensive set of benchmarks to evaluate storage system technologies offered by vendors. Our benchmarks are designed in a way that they mimic the realistic I/O workloads and also allow integrated and traditional block-based storage solution providers to bid on our RFP.

One of the biggest challenges in tape archiving lies in the area of media refreshment. While replacing, updating, or increasing the amount of front-end disk cache or servers responsible for data movement to/from disk and/or tape is a relatively straightforward and non-intrusive process, the process of media refreshment presents many challenges. In a large-scale tape archive such as that at the OLCF, where 10s of thousands of individual tape cartridges are managed, at any given time there may be thousands of tapes housing multiple PBs of data needing to be retired from service. Unfortunately, the data on those tapes no longer resides on disk cache in most cases, and must be read from the older tapes in order to be written to newer media. Under real life conditions, where resource constraints such as utilization on data mover server(s) and the number of drives available to mount such media are a reality, the process of refreshing older media can literally take years. For example, here at the OLCF we are actively retiring 10,000+ 9840B tapes from service, and based on the performance to date, we expect that process to continue for the next 2.5 to 4 years. The OLCF has recently purchased a small quantity of 9840D drives so we can read the 9840B tapes at a 30% faster rate—in order to bring us closer to the 2.5 year figure. While that process is underway, we are simultaneously retiring several thousand 9940 tapes from service, and that initiative is expected to take approximately one year to complete as well. Media refresh(es) will continue to be a "day-to-day" operation going forward. For purposes of planning and procurement, it is assumed that 5-10% of total HPSS system resources will be utilized for media refresh operations.

THE RELIABILITY AND AVAILABILITY OF STORAGE SYSTEMS

The OLCF tracks a series of metrics that reflect the performance requirements of DOE and the user community. These metrics assist staff in monitoring system performance, tracking trends, and identifying and correcting problems at scale, all to ensure that OLCF systems meet or exceed DOE and user expectations.

$$\mathrm{SA}=\left(\frac{\text{time in period}-\text{time unavailable due to outages in the period}}{\text{time in period}-\text{time unavailable due to scheduled outages in the period}}\right)\times 100$$

Scheduled Availability (SA) measures the effect of *unscheduled* downtimes on system availability. For the SA metric, scheduled maintenance, dedicated testing, and other scheduled downtimes are not included in the calculation. The SA metric is to meet or exceed an 85% scheduled availability in the first year after initial installation or a major upgrade, and to meet or exceed a 95% scheduled availability for systems in operation more than 1 year after initial installation or a major upgrade. Reference Table 1.

Table 1. OLCF Computational Resources Scheduled Availability (SA) Summary 2010–2011

System	CY 2010		CY 2011 YTD (Jan 1-Jun 30, 2011)		
	Target SA	Achieved SA	Target SA	Achieved SA through June 30, 2011	Projected SA, CY 2011
HPSS	95%	99.6%	95%	99.9%	>95%
Spider	95%	99.8%	95%	98.5%	>95%
Spider2	N/A	N/A	95%	99.9%	>95%
Spider3	N/A	N/A	95%	99.9%	>95%

Table 2. OLCF Computational Resources Overall Availability (OA) Summary 2010–2011

System	CY 2010		CY 2011 YTD (Jan 1-Jun 30, 2011)		
	Target OA	Achieved OA	Target OA	Achieved OA through June 30, 2011	Projected OA, CY 2011
HPSS	90%	98.6%	90%	98.9%	>90%
Spider	90%	99.0%	90%	96.5%	>90%
Spider2	N/A	N/A	90%	99.1%	~99%
Spider3	N/A	N/A	90%	99.2%	~99%

$$\mathrm{OA}=\left(\frac{\text{time in period}-\text{time unavailable due to outages in the period}}{\text{time in period}}\right)\times 100$$

Overall Availability (OA) measures the effect of both *scheduled and unscheduled* downtimes on system availability. The OA metric is to meet or exceed an 80% overall availability in the first year after initial installation or a major upgrade, and to meet or exceed a 90% overall availability for systems in operation more than 1 year after initial installation or a major upgrade. Reference Table 2. As indicated by these numbers, both HPSS and our Spider file systems provide extremely high availability. Overall availability of these systems continues to dramatically exceed our operational requirements. The decrease in overall availability in one of our Spider file systems in 2011 compared to 2010 was due to an increase in the number of dedicated system times taken to evaluate new features and stabilize the next Lustre release. Spider2 and Spider3 remained available during these dedicated system times thereby minimizing impact to users.

Within HPSS, DB2 is used as the storage mechanism for all file/device metadata (ownership, status, location, etc.). DB2 has been proven in the field over many years and is well known for its reliability and availability features.

The front-end disk cache for the HPSS tape archive is comprised of several RAID6 arrays, with individual LUNS "owned" by mover servers responsible for data flow to/from disk. Currently, in our configuration here at the

OLCF, each mover has a single FC or IB path to a target LUN, but we are actively working on modifying that configuration in order to provide multipathing for our disk cache.

HPSS has the ability to store data on multiple levels of tape if so desired. Here at the OLCF, by default, data is written to one level of tape when migrated from the front-end disk cache. Users have the option of specifying a different "Class of Service" in order to have their data written to two levels of tape—providing an extra level of protection in case a media problem is encountered. Due to cost concerns, that is only encouraged and/or recommended for critical data.

While currently not in use at OLCF, HPSS does have High Availability capabilities based on Red Hat Linux cluster services. In this model, HPSS can provide failover redundancy for critical HPSS components—core server, data movers, and gateway nodes.

A feature that will soon be incorporated into HPSS is RAIT–Redundant Array of Independent Tape. RAIT will provide an additional level of redundancy and fault tolerance related to media failures without suffering the full cost penalty associated with the traditional method of having data on more than one level of tape.

Maintenance Activities

Maintenance activities for the Spider file systems are planned for once per quarter and planning for the "next" maintenance window begins shortly after the "previous" maintenance ends. It starts with a post-mortem analysis of the previous maintenance, and then developing a list of items to perform. At ~2 weeks pre-outage tasks are capped for the upcoming maintenance. A full outage plan is developed including any dependencies that the Lustre team has on other teams inside HPC Operations. This plan is documented on the internal wiki, and is shared through several normally scheduled weekly meetings as well as any outage/maintenance prep meetings. Coordination with the Facilities group is also necessary if one of the reasons for the outage is work being done to the power or cooling infrastructure. This planning process helps us to document upcoming changes/modifications, record their completion date, and also learn from issues that may come up during the maintenance – making the next maintenance hopefully smoother. They also help to enforce overall system knowledge in the administrative team and enforce, through the evaluation of the planned steps, a best practices approach to system administration.

Data Integrity

For Spider the RAID protections are the only data protections that are in place system wide. Applications can choose to add data protections in their simulation and modeling, but we've found that if we enforce anything it hinders performance and may not be what the application needs. End-to-end checksums are currently under evaluation for the next-generation Spider deployment.

End-to-end checksums is a feature recently introduced to HPSS. While not currently in use at OLCF, checksum utilities allow a user to perform a checksum of file content and place the results in a User Defined Attribute for later comparison if/when the file is retrieved [6]. At this time at the OLCF, individual users/departments in some cases perform pre and post retrieval checksums in order to verify data integrity.

24 x 7 Support Model

In support of a 24x7 operation, we use Nagios to monitor the correct configuration of the file system, and either through SMS messaging or direct phone calls from the 24x7 Computing Operations Center, notify the on-call administrator for the system of any critical event that causes availability to be degraded or lost. In the event of facility events during off hours, the Operations Center will call the HPC Operations Group Leader and then will notify affected teams. In the case of the Lustre team, scripts have been developed to quickly put the DDN controllers into power saving mode, power down the OSSes, MDSes, etc. to lower the heat load in the room. The DDN S2A9900 controllers have a disk sleep mode that parks the heads and spins down the disk.

THE USABILITY OF STORAGE SYSTEMS

Conventional methods for addressing I/O bottlenecks, such as increasing I/O backend capability by adding more disks with higher speeds, are unlikely to keep up with the performance issues due to the costs associated with storage. The problem is further exacerbated by the inefficiency of I/O performance; some applications are unable to achieve a significant fraction of the peak performance of the storage system. This can be due to a variety of factors the complexity of traditional I/O methods, where the developer has to make a heroic effort to optimize the application I/O. This limit on usability directly impacts the possible performance of the application. The OLCF has implemented a multi-point approach to addressing these challenges.

The ADIOS I/O framework was designed with the aforementioned concerns in mind. The ADIOS I/O framework [9] not only addresses the system I/O issues, but also provides an easy-to-use mechanism for the scientific developers to work from I/O skeleton applications. Through the use of an optional metadata file, which describes the output data and enables automatic generation of the output routines, the burden on the user is substantially reduced. ADIOS componentizes different methods of I/O, allowing the user to easily select the optimal method. In concert with data staging, this work exemplifies a next generation framework for I/O.

As common in many next-generation software projects, the the biggest challenge is often one of technology adoption, that is, getting users to change from current I/O implementations to ADIOS. As the ADIOS ecosystem continues to grow, we believe that ADIOS will gain a wider spread acceptance.

ADIOS and our eSiMon dashboard are used by the combustion, climate, nuclear, astrophysics, and relativity communities. In particular we have created a I/O skeleton generation system, using ADIOS, and have applied this in 10 applications, to make it easy for computing centers to analyze I/O performance from many of the leading LCF applications, with virtually no working knowledge of each application on their systems.

ADIOS has worked well with all current users, and have often shown over a 10X improvement of using other I/O implementations; see Figure 2 for I/O performance of the S3D and PMCL3D simulations on the Jaguar system.

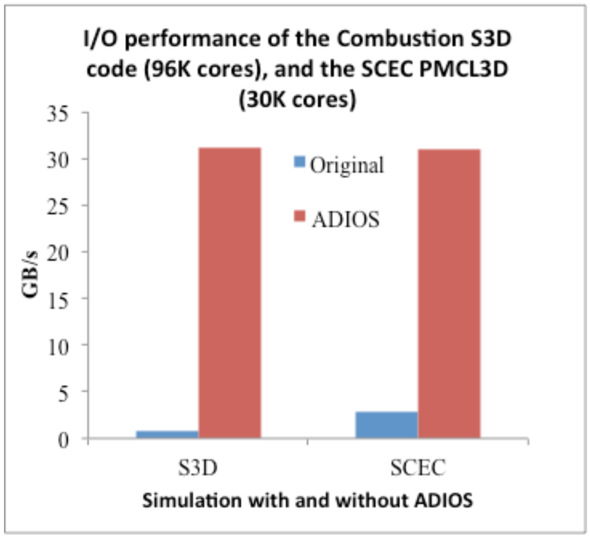


Figure 2. ADIOS performance.

CONCLUSIONS

Oak Ridge Leadership Computing Facility has developed extensive developed key competencies in architecting and administration of large-scale Lustre deployments as well as HPSS archival systems. Lessons learned from past Lustre and HPSS deployments and upgrades help us to better adopt to changing technology and user requirements.

REFERENCES

1. Shipman, G; Dillow, D.; Oral, S.; Wang, F. *The Spider Center Wide Filesystem; From Concept to Reality*. In Proceedings of Cray User's Group 2009.
2. OpenSFS. http://www.opensfs.org
3. Nagios. http://www.nagios.org.
4. Lustre Monitoring Toolkit (LMT). https://github.com/chaos/lmt/wiki.
5. CFengine. http://www.cfengine.com
6. Extreme Scale Storage for a Smarter, Faster Planet http://www.hpss-collaboration.org/documents/HPSSBrochure.pdf
7. Schlep, F. *The Guide to Better Storage System Operation*. Indiana University Press, Bloomington, IN, USA, 1973.
8. [Polte2009] Polte, M., Lofstead, J., Bent, J., Gibson, G., Klasky, S.A., Liu, Q., Parashar, M., Podhorszki, N., Schwan, K., Wingate, M. and others. "...And eat it too: High read performance in write-optimized HPC I/O middleware file formats". Proceedings of the 4th Annual Workshop on Petascale Data Storage, pp 21-25, 2009.
9. Lofstead2009] J. Lofstead, F. Zheng, S. Klasky, K. Schwan. Adaptable, Metadata Rich IO Methods for Portable High Performance IO. Parallel & Distributed Processing, IPDPS'09, Rome, Italy, May 2009, DOI=10.1109/IPDPS.2009.5161052.
10. SEC - Simple Event Correlator: http://simple-evcorr.sourceforge.net/.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Dominik Ulmer
CSCS
dulmer@cscs.ch

Stefano C. Gorini
CSCS
gorini@cscs.ch

ABSTRACT / SUMMARY

The Swiss National Supercomputing Center CSCS has introduced a business model which turns data services from a reactively to a pro-actively managed service. A clearly defined center-wide file system hierarchy in conjunction with a set of specialized computers for data analysis allows to optimize storage systems characteristics like bandwidth or latency for different systems and workloads, to plan and manage capacities within the resource allocation process for computing time, and to leverage technical and financial synergies between the different service categories. Storage services are based on Luster and GPFS software with TSM/HSM extensions. A combination of different storage hardware technologies like SATA, SSD, and tape are used for the services depending on the individual requirements.

INTRODUCTION

For many years, data was a side-business to computing for HPC centers. Large-scale storage systems were architected and installed as peripherals of a supercomputing procurement. However, data growth rates exceed performance growth rates of HPC systems and therefore storage systems become an increasingly more significant part of the investment and operational budget of the computing centers. While the Swiss National Supercomputing Centre CSCS recognizes the importance of data for computational sciences, it does not have the intention to turn from a high-performance computing center to a data storage and management center. It is therefore essential to understand the role of data in the workflow of computational scientists using supercomputers and to accordingly architect the data services offered by the center.

SYSTEM BOUNDARIES

The main function of a HPC center is to enable computational scientists to use supercomputers for their research. This involves the preparation and the processing of large data sets, either for preparing input for computing runs or for analyzing data, which may be both, measured data or the result of a computational job. In both cases, one deals with living data for ongoing research projects, i.e. a time span that is well below 10 years. Long-term archiving for documentary purposes is not in the core business of a supercomputing center. A data service at a HPC center in this framework has to address the following topics:

- support of the computational workflow by means of an integrated architecture of computing, storage, and data analysis systems
- a storage hierarchy which is easy to understand by the user and provides a clear basis for the management of technical requirements
- a business model that allows the center to plan investments and operational costs in advance and which is aligned with the business model for providing computational resources.

THE CSCS STORAGE HIERARCHY

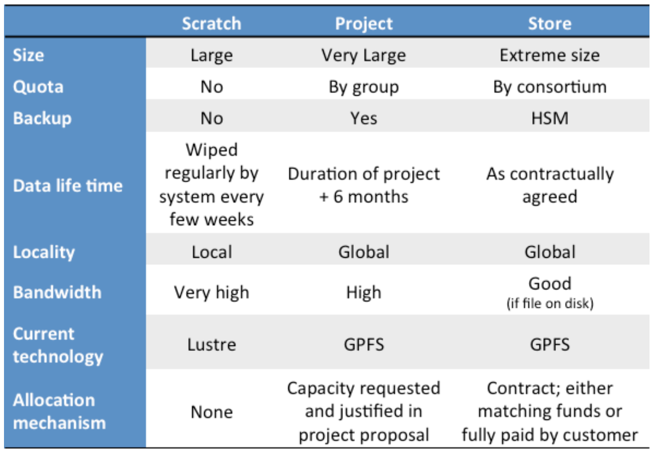
CSCS distinguishes three different levels of storage (see **Error! Reference source not found.**):

A) SCRATCH file system

The purpose of the scratch file system is to provide a storage container for running an individual computational job resp. an individual suite of computational tasks. Data remains only temporarily on the file system and must be copied to a different storage level for permanent storage. The file system has no quotas for user or groups. Old files are automatically deleted in order to maintain capacity. The scratch file system is local to an individual computer and its technical characteristics are specified according to the architecture of the system and the expected workload.

B) PROJECT file system

The project file system provides a data management and storage space for an individual computational project. CSCS issues a call for project proposals twice a year. Researchers can request computational and storage resources in their proposals, which are evaluated by an external committee with respect to their scientific quality and impact. The size of the storage request and of the compute cycle request must be justified in the proposal and must be coherent to each other. The project receives a storage quota which is shared between all members of the project team. The project file system is globally mounted on all CSCS user facilities and provides enough bandwidth for efficiently transferring large data sets to and from the scratch file systems. It provides extended user functionalities like snapshots. Data is kept on the project file system for the duration of the computational project (up to 3 years) plus 6 months in order to allow the user transferring the final data to a longer-term storage system or to the storage resource of a successor project.



	Scratch	Project	Store
Size	Large	Very Large	Extreme size
Quota	No	By group	By consortium
Backup	No	Yes	HSM
Data life time	Wiped regularly by system every few weeks	Duration of project + 6 months	As contractually agreed
Locality	Local	Global	Global
Bandwidth	Very high	High	Good (if file on disk)
Current technology	Lustre	GPFS	GPFS
Allocation mechanism	None	Capacity requested and justified in project proposal	Contract; either matching funds or fully paid by customer

Figure 1: Hierarchy of file systems at CSCS

C) STORE file system

Large research projects are often carried out by consortia, which combine many research groups and projects as identified by the CSCS call for proposal process. Research consortia share data between the individual projects and teams and they manage the data sets over a longer timespan.

CSCS offers the store file system for such consortia. In contrast to the scratch and project file systems, resource on /store is not for free, but requires a financial contribution. Up to a certain limit, academic consortia can get storage space on store on the basis of matching funds. Above the limit and for non-academic consortia, direct investment and operational costs must be fully paid. A consortium must describe its overall research plan and goals, in order to assess the strategic importance of the consortium to science and the HPC center and to define the duration of the contract.

/store is a global file system that can be accessed from all user-accessible computers at CSCS. As it is based on a hierarchical storage management system, which is to a large extent based on tape, bandwidth is lower than to the project file system.

TECHNICAL IMPLEMENTATION

All three storage levels are built with parallel file system technology in order to ensure performance scalability.

The scratch file systems are currently mainly based on Lustre, which allows for optimal read/write performance. Stability is sufficient and enhanced functionalities are not required because of the shared nature of the file system. Both, LSI and DDN storage controllers have been deployed for different implementations, mainly as direct attached scratch. Because of the meta-data performance bottlenecks in the current Lustre architecture, SSDs have been successfully tested for improving meta-data performance, although the fundamental problem of a non-distributed meta-data store can only be eased but not completely resolved with this approach.

The project file system is characterized by the combination of parallel HPC-type file system features with some enterprise storage requirements. It must be able to handle a large number of files with very good meta-data performance and has to offer functionalities like quota, snapshots, and integration with backup software. CSCS uses very similar storage hardware as on the scratch file systems, driven from separate storage servers that are connected to a high-speed Infiniband network backbone. GPFS has been selected as software technology for this file system because of its RAS features but also because superior meta-data performance compared to Lustre.

For the store file system, raw I/O performance is not as important as for the other two file systems. Technical and financial analysis showed that it is easily implemented with the same GPFS technology as /project combined with the TSM/HSM product of IBM. The TSM solution at CSCS also includes a backup and disaster/recovery functionality which enables us in the case of the total loss of the file system to recover all GPFS metadata within a few hours and all critical files within two days. By sharing licenses, infrastructure, and knowhow, operational costs can be kept low.

CSCS would be interested to change from the proprietary GPFS technology to open-source/public domain software. Lustre in its current state does not seem to be a viable option. If Lustre will be developed further in a coherent fashion, with stable funding and a clear roadmap, it could be envisaged to use in the future Lustre as the fundamental file system technology in combination with pNFS for mounting non-HPC clients.

As described above, we consider data analysis systems as an integral part of a data service. CSCS has decided to offer a portfolio of different computer architectures for data analytics: a standard, fat node cluster; a GPU cluster; a large-shared memory system based on the SGI Altix UV architecture; and a massively-multithreaded Cray XMT2 system. Access to these systems is granted within the allocation process for computing time on the main HPC systems.

CONCLUSIONS

By defining a coherent business model for data and storage, the Swiss National Supercomputing was able to simultaneously optimize costs and scientific workflow at the center. For long-term sustainability, however, users additionally have to be educated to rethink their storage needs and patterns by means of in-situ data analysis and rewriting the I/O in their codes.

CSCS considers IBM's GPFS technology to currently be the most advanced solution for highly available and powerful *global* parallel file systems. We use GPFS with its characteristics of an enterprise file system only for the global levels of the storage hierarchy. Thus, the number of required client licenses can be drastically reduced by using a small number of I/O forwarding nodes per system. The bandwidth-hungry local scratch file systems, in which every compute node is a client, are built with Lustre. Solid-state memory technologies have developed into a viable alternative or addition to storage hardware solutions, boosting latency and IOPS-sensitive components of the storage system to new performance levels.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

HPC Enhanced User Environment (HEUE) Position Paper

Thomas M. Kendall
U. S. Army Research Laboratory
DoD High Performance Computing Modernization Program
Thomas.m.kendall4.civ@mail.mil

Cray J. Henry
DoD High performance Computing Modernization Program
cray@hpcmo.hpc.mil

John Gebhardt
Lockheed-Martin/U. S. Air Force Research Laboratory
DoD High performance Computing Modernization Program
John.gebhardt@lmco.com

ABSTRACT / SUMMARY

The DoD High Performance Computing Modernization Program (HPCMP) is now implementing a major change to at all its DoD Supercomputing Resource Centers (DSRC) through the introduction of a center-wide file system (CWFS) and an integrated life-cycle management hierarchical storage manager (ILM HSM).

Following discussions with its top consumers of archival capacity, the HPCMP architected a strategy to enable its customers to reduce archival requirements. The key elements of the HPCMP's strategy are:

- Provide tools to enable customers to associate project specific metadata with files in the archive; enable automated scheduled actions keyed against specific metadata; enable users to control second copy behavior; and enable user specified logical data constructs suitable for building case management features.
- Provide an intermediate level of storage between the HPCMP's traditional two tier scratch and archive architecture. This intermediate storage (i.e. CWFS) will enable customers sufficient time to analyze results, and archive analysis results rather than 3-dimensional restart files. The center-wide file system is sized to allow 30 days of analysis before transfer to archive.
- The introduction of the center-wide file system also creates the opportunity to enhance and upgrade interactive customer support with high performance graphics and large memory to support the analysis efforts.

INTRODUCTION

The HPCMP built forecasts of future archival storage capacity needs based upon past history and concluded that the current growth rate was unsustainable. Left unchecked, storage would consume the majority of the HPCMP's budget by the end of the decade. A key finding of the subsequent analysis was that the archive costs remained in an affordable range if the archive growth rate was constrained to 1.4 times the growth of the previous year's growth. This finding is tied to an assumption that industry doubles tape capacity every 24 months. If the growth rate of the archive exceeds the rate of tape capacity increase, the HPCMP has to fund tape libraries, slots, and potentially licensing for the additional capacity.

After gathering input from the principal investigators of the projects that consumed the vast majority of the program's archival capacity, several recurring themes emerged:

- Existing storage tools were insufficient to manage large datasets and the use of filenames to capture relevant metadata was no longer practical.
- Raw computational outputs were being archived due to insufficient analysis time for data stored in scratch space.
- Performing analysis using batch resources was adding to the problem of insufficient time for analysis.

These observations were further vetted and ultimately formed into requirements for the HPCMP's next generation storage solution. A working group, with representatives from the HPCMO, the DSRCs, and user advisory groups, was formed. The group was chartered to further develop and refine the requirements and to develop the architecture for data flow within the HPCMP.

The architecture that the storage working group arrived at included a combined information lifecycle management and hierarchical storage management layer.

A subsequent effort surveyed the information lifecycle management and high performance storage markets, leading to the creation of an acquisition strategy. A key element of this strategy was the separation of hardware and software requirements and provisioning.

A market survey determined, to no great surprise, that a mature information lifecycle management solution integrated with hierarchical storage management did not exist. The strategy that emerged was to seek a partnership with industry aimed at fostering the integration of a leading information life cycle management solution with a leading hierarchical storage manager. Our software requirement allowed for an initial capability that could evolve into a fully integrated solution over 10 years.

The combined ILM HSM requirement was called "HPCMP Storage Lifecycle Management." A Request for Proposals was released in March of 2009 and a contract awarded in August of 2009.

With the ILM+HSM addressing the software requirements for improved tools to manage data, the remaining primarily hardware requirements were for the Center Wide File System and the Utility Server. This second component of the acquisition strategy was focused on the required hardware to deliver the new services.

Much of the requirements for the Center Wide File System (CWSF) and utility server derived from the winning ILM+HSM solution. Subsequently, two Requests for Proposals were issued. The RFPs required responses for the six DSRC locations and for a range of file system capacities and performance levels. They also required the inclusion of a 10 gigabit network fabric for connection of the Center Wide File System components, the utility server nodes, and the HPC system login nodes at each DSRC. The storage capacities requirements ranged from 250 TB to 2 PB. The I/O performance requirements ranged from 8.0 to 40.2 GB/s and from 70,000 to 320,000 file open/creates per second.

Awards for the CWFS and utility server were made by Lockheed-Martin in September 2010 and deliveries were completed in December, followed by acceptance and integration. The systems were transitioned into production sequentially center by center between June and August 2011.

In 2011, the HPCMP also took steps to refresh its tape archive hardware. Based on the earlier analysis showing that a doubling of tape capacity every 24 months was a key component of cost containment, the program compared commodity LTO drives with the proprietary Oracle T10000 line. Although the LTO family drives have a lower initial purchase price than the T10000 family drives, the lifecycle costs for LTO were found to be significantly higher. Drivers were the need to replace the media with each generation of LTO drive in order to realize the increase in capacity and the slightly slower capacity growth rate (15x over ten years for LTO and 10x over five years for T10000).

CONCLUSIONS

It is too early to gage the impact on the growth of the HPCMP's archive as a result of the acquisition and deployment of the HPCMP Enhanced User Environment. The Program's advisory bodies have responded positively to the goals and progress. The initial feedback for the additive analysis capability provided by the utility server and Center Wide File System has been positive. Once the Storage Lifecycle Management solution is fully deployed for production use in October 2011, the full effects of HEUE will be measured and reported.

In terms of the adoption of commodity hardware, the tape industry would need to seek ways to reduce the frequency of complete media replacements while meeting or exceeding the 1.4 compound annual capacity increase target.

U.S. Department of Energy Best Practices Workshop on
File Systems & Archives
San Francisco, CA
September 26-27, 2011
Position Paper

Stephan Graf
Jülich Supercomputing Centre
st.graf@fz-juelich.de

Lothar Wollschläger
Jülich Supercomputing Centre
l.wollschlaeger@fz-juelich.de

ABSTRACT / SUMMARY

The storage configuration for the supercomputer *JUGENE* in Jülich consists of a GPFS cluster (*JUST*) and two Oracle STK SL8500 tape libraries. In this paper the actual configuration and the next upgrades are described. Furthermore a project for using flash storage as a kind of cache memory for the disk storage is introduced.

INTRODUCTION

The Jülich Supercomputing Centre (JSC) operates two supercomputer: The BlueGene/P System *JUGENE* and the x86 based *JuRoPA* system. While the *JUGENE* uses the remote GPFS cluster *JUST*, the *JuRoPA* users works on a local Lustre based storage. There the users can access their files in the GPFS file system via dedicated nodes.

The consideration in this paper for the actual and the future storage configuration/implementation are focused on the *JUGENE* and the *JUST* GPFS cluster.

The users can access three types of file systems:

On $HOME they should store there code and develop their program.

For the job run they are urged to use the scratch file system $WORK to get the maximum IO performance.

To archive their results the data should be moved to the $ARCHIVE file system.

JUGENE STORAGE PERFORMANCE TODAY

The *JUGENE* is build up of 72 BlueGene/P Racks with 1 PF peak performance and 144 TiB main memory. Each rack contains 1024 compute nodes (CN) and 8 IO nodes (576 IO nodes in total), with each one connected via 10GbE to the *JUST* storage. Measurements show that a single IO node gets an IO performance of 450 MB/s reading and 350 MB/s writing. For a whole rack it is 3.6 GB/s reading and 2.8 GB/s writing. The maximal peak IO for the full system is 260 GB/s reading and 200 GB writing. Assuming that 50% of the main memory of one rack (1024 CN) is to be written on file system (e.g. for checkpointing), the required time is 5 minutes for reading and 7 minutes for writing. To write 50% of the main memory of the full system in 15 Minutes requires 0.5 *144 TiB /1800s = 44 GB/s. The *JUST* cluster based on DS5300 storage devices provides 66 GB/s. But only half of the cluster is used for the fast scratch file system $WORK. The other half of the clusters hosts the $HOME and $ARCHIVE file system. This implicates that the $WORK can be saturated by 33 GB/s / (8*0.35 GB/s) = 12 racks writing to the file system.

On the *JUST* cluster 8 building blocks provides the $WORK file system, each containing a DS5300 with 36 LUNs per DS5300 having a size of 8 TB (RAID6). This leads in a total capacity for $WORK of 2.3 PB.

BLUEGENE/Q INSTALLATION IN 2012

In 2012 the *JUGENE* will be replaced by a BlueGene/Q system consisting of 6 racks. There are 8 IO nodes per rack, each having a dual 10GbE port with an aggregated bandwidth of 1.5

GB/s. So the maximum throughput of a rack is 12 GB/s and the full system 72 GB/s.

If 50% of the main memory of on rack (1024 CN with 16 GB RAM per node) are to be written on disk it will last (approximately) 12 minutes. So to write 50% of the full system main memory to the storage in 15 minutes, a bandwidth of 50%*384 TiB /1800s = 115 GB/s are required. Therefore we will get a storage upgrade for the *JUST* GPFS cluster. We are planning to install 8 DDN SFA12000 and getting an aggregated bandwidth between 100GB/s and 160 GB/s for the scratch file system $WORK. The performance for $HOME and $ARCHIVE will also increase, but this is not concerning us.

FLASH MEMORY AS SCRATCH FILE SYSTEM

In parallel the JSC will investigate a new storage concept using flash memory cards as a kind of cache between the IO nodes and the ordinary disk storage. It is a European Union funded project, a PRACE (Partnership for Advanced Computing in Europe) prototype for next generation supercomputers.

4 x86 systems each with 2 fusionIO ioDrive Duo 320GB SLC will be set up. The bandwidth of the flash card is 1.5GB/s. The cumulated performance of these 4 nodes should be 12 GB/s, a similar value as 8 BlueGene/Q IO nodes (one rack). Using the GPFS features to setup different kind of storage pools and to implement placement and migration policy rules a concept will be modeled, that new created files will be created on flash, and GPFS will migrate the files to disk in the background automatically.

This concept will be implemented with real BlueGene/Q hardware as soon as it is available.

ARCHIVE STORAGE EXPANSION

The users should store their results on the $ARCHIVE file system. There the data will be migrated by Tivoli HSM on tapes (weighted by *size* and *last access time*). For safety two versions (COPYPOOL) will be held on tape. Furthermore every file of the $HOME and the $ARCHIVE file system will be backed up. Files on the scratch file system $WORK are not backed up and will be deleted after 90 days.

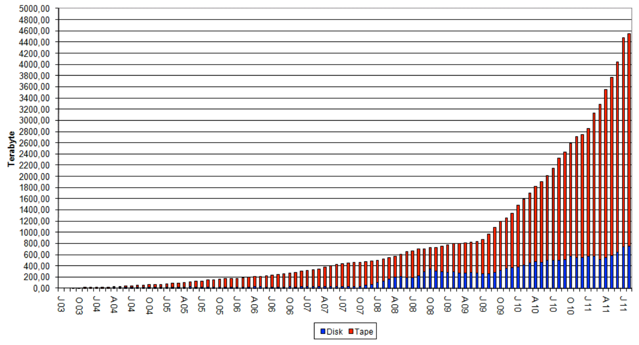


Figure 1: Data groth on the GPFS cluster JUST

The JSC operates two Oracle STK SL8500 libraries with an aggregated capacity of 16.6 TB (T10K-B tape drives). In figure 1 the exponential data growth on our storage cluster can be seen. We expected to run out of space in the third quarter 2011. Because of the ordering and shipping delay of the new hardware it became critical the last month. But now the new hardware has arrived and is going in production. 16 T10K-C tape drives have been added and the new tape generation (which is able to store 5 TB) will replace the old tapes step by step.

This kind of upgrade is the typically way for us to manage the growth of data amount. For the next 6 years we plan to enlarge the capacity of the two libraries to 80 PB just by upgrading to the next tape drive generation T10K-D.

CONCLUSIONS

On our supercomputer a specific maximal I/O performance is available and for the user it is reasonable to get the maximum performance from the file system. But this is often difficult to achieve. Therefore it is mandatory to train the users and give them the knowledge to speed up their jobs I/O. For this purpose we have developed the *SIONlib* in Jülich. The users can use this library in there code to map very easily local task I/O to one file. By using the *SIONlib* it is possible to get nearly 100% of the performance on the scratch file system $WORK from the JUGENE. We also use this tool for benchmarking parallel file systems.[1]

The other subject concerns the long time data storing. Till now and for the next years we are able to store all user data in our archive system. The new technologies keep up with the data growth in Jülich. But there are upcoming questions like how long must the data be hold or what happens when a project ends. These problems must be tackled in mid or long term.

REFERENCES

. [1] http://www.fz-juelich.de/jsc/sionlib/

U.S. Department of Energy Best Practices Workshop on File Systems & Archives
San Francisco, CA
September 26-27, 2011

Andrew Uselton
NERSC/LBL
acuselton@lbl.gov

Jason Hick
NERSC/LBL
jhick@lbl.gov

ABSTRACT / SUMMARY

This position paper addresses the business of storage systems and practices related to planning for future systems (I-1A) and establishing bandwidth requirements (I-1B), with some discussion also relating to the administration of storage systems and the monitoring of specific metrics (II-2A). The best practice is to balance I/O with compute capability.

We present a quantitative characterization of "HPC and I/O system balance" by examining the relative costs of compute resources and I/O resources on the one hand and the relative impact of compute and I/O activities on the other.

INTRODUCTION

An HPC system with too little I/O infrastructure to support its workload could leave much of the compute resource idle as it waits for I/O operations to complete. The idle compute resource represents an opportunity cost in that it may have no other useful work to do during the wait.

BACKGROUND

One study [2] suggests that memory capacity is the key determinant of necessary I/O bandwidth and capacity. Figure 1 presents a traditional guideline for balancing I/O.

The relationship between performance and memory comes from the need to flush the contents of memory to persistent local storage in a combination of reasonable time and cost.

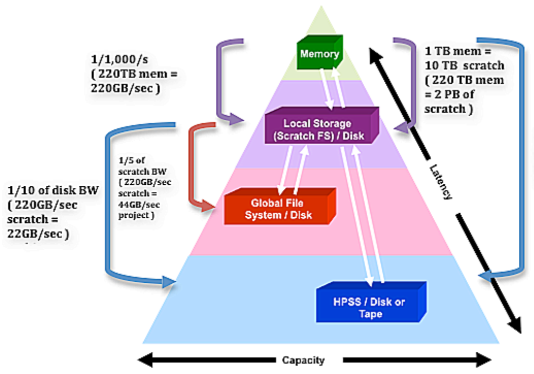

1/1,000/s
(220TB mem = 220GB/sec)
1/10 of disk BW (220GB/sec scratch = 22GB/sec)
1/5 of scratch BW (220GB/sec scratch = 44GB/sec project)
Memory
Local Storage (Scratch FS) / Disk
Global File System / Disk
HPSS / Disk or Tape
1 TB mem = 10 TB scratch (220 TB mem = 2 PB of scratch)
1 TB mem = 35 TB new data per year (220 TB mem = 7700 TB of data)
Latency
Capacity

Figure 1. Conventional HPC I/O Planning Guidelines

Additional I/O resources provide diminishing returns, so there is a point of balance at which bandwidth is "just enough", and in this case the heuristic is to move all of memory in about 1000 seconds.

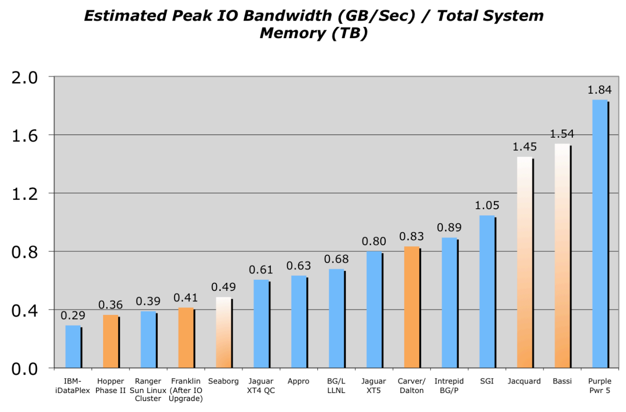


Figure 2. Peak bandwidth to system memory

Figure 2 presents this heuristic as applied to several HPC systems. By that metric, systems with a value over 1.0 have over-provisioned I/O subsystems relative to system memory capacity. A subjective review of such systems reveals that users are happy with the I/O bandwidth they deliver.

The purpose of this paper is to propose an alternative characterization of balance using a cost-based model in conjunction with the compute and I/O workloads of the HPC system. As a starting point, this discussion abstracts away much of the complexity to arrive at some core ideas.

SYSTEM BALANCE

As a first simplifying assumption, suppose that the cost of an HPC system is composed entirely of the budget for the compute capability and the budget for the I/O capability. Next, suppose that the work produced by an HPC system is measured as the number of jobs completed weighted by the size of each job in two dimensions: the number of *node-seconds* used in the computation and the number of *node-seconds* used in I/O.

Further, assume that the aggregate compute capability is near linear in the cost of the of compute nodes:

$$C(n) = M_n \times n$$

where M_n is the marginal cost of nodes. Similarly, assume the aggregate I/O capability (measured as its peak rate) is near linear in the cost of the I/O infrastructure:

$$I(r) = M_r \times r$$

where M_r is the marginal cost of adding a unit of bandwidth r.

Now let the utilization U of the HPC system be given by the fraction of *node-seconds* spent on compute activity, given a particular workload. Our characterization of "system balance", given n and r, is given by:

$$B = \left(\frac{U}{(1-U)}\right)\left(\frac{I(r)}{C(n)}\right)$$

Our claim is that at $B = 1$, the system is in balance in that it achieves the maximum amount of workload per dollar spent. As an example, if you spend 10% of your HPC system budget on I/O infrastructure ($\frac{I(r)}{C(n)} \cong 0.1$), then the nodes should be spending 10% of their time on I/O, and the rest on computation ($\frac{1-U}{U} \cong 0.1$).

This is a relatively intuitive idea given the simplifying assumptions, but it begs the question, "What is B on my system, given its workload?" Those who design and purchase HPC systems are very familiar with the total cost and the fraction spent on I/O infrastructure. On the other hand, it is not at all clear what the value of U is. It will certainly be different at different times and for different workloads. We propose that monitoring the jobs and I/O on the system for any given day's activity and for longer intervals will yield the value of U.

CHALLENGES

Some system designs and I/O strategies attempt to improve I/O performance by departing from this simple model. For example, a strategy that overlaps computation and I/O will yield a higher utilization. In that case it becomes important to estimate both the expected impact and the extent to which the strategy is implemented in practice. If a strategy can entirely "hide" I/O activity but only affects 10% of the workload, then the simplified model is still close to correct.

The model has plenty of room for improvement. For example, the cost model does not need to as simple as presented. There may be fixed costs and nonlinearities, and the model could incorporate them without difficulty. The model can also include other aspects of HPC system architecture, for example, adding *node-seconds* spent in (node to node) communication. In some cases that communication will compete for bandwidth with the I/O requirements, leading to additional complexities.

CASE STUDY

The *Carver* IBM Dataplex cluster at NERSC was provisioned with approximately 15% of its budget dedicated to I/O infrastructure. The system has 30 TB of memory suggesting a target bandwidth to storage of 30 GB/s using the heuristic from the background discussion. *Carver's* measured bandwidth is about 25 GB/s, so it is designed to be at about 83% of that target. *Carver* runs the Integrated Performance Monitoring (IPM) library [3] with every scheduled job. Each job produces a report at the end of its execution giving the time spent in computation, the time spent in I/O, and the amount of data moved (among other quantities). IPM provides a comprehensive profile of compute and I/O activity for a given interval. From that profile it is possible to directly calculate the utilization. For example, in June 2011 $U \cong 0.94$. The balance factor for the actual workload is around 2.5. By this measure, the system's balance favors I/O and could handle a heavier load.

CORRELATING I/O ACTIVITY WITH JOBS

Most HPC systems do not have IPM or other direct measures of the utilization. Without that information we do not know what balance has been achieved in practice after having applied the heuristics from Figure 1. An alternative strategy is under development at NERSC that infers the utilization U from server-side I/O monitoring with the Lustre Monitoring Tool (LMT) [4]. Server-side data is anonymous with respect to the nodes that generate the I/O. Nevertheless, it is often possible to infer the job from the I/O pattern. When that can be done comprehensively it will yield the utilization as before, and therefore give a quantitative gauge of the balance.

On NERSC's *Franklin* Cray XT4 there are commonly more than one hundred jobs running at a given time, and the I/O workload resulting from that compute workload is potentially composed of I/O from many jobs simultaneously. Often, an application runs many times repeating the same I/O pattern each time. From that collection of jobs (call it a *job class*) we calculate the average I/O behavior for the application, which is an approximation of its expected behavior in isolation from other jobs. The individual calculated behavior of each of the whole suite of applications provides the initial estimate for the behavior of the system as a whole, and the estimates can be iteratively refined via a generalized linear regression. This is a computationally expensive task but straight forward, in principle.

As an example, we examine the IOR [5] file system benchmark, which runs as a regularly scheduled test of the *Franklin* scratch file systems. 175 such tests were run in July 2011, and the system job log records the start and stop time of each job along with the number of nodes used. The IOR application runs using the same parameters in order to provide a repeatable health-check of the file system. Each job writes *4GB* to the file system from each of 64 tasks on 16 nodes. It then reads that data back in. Jobs generally run for 150 to 200 seconds, but can run much longer when the file system is occupied with other I/O. The jobs are submitted to an I/O-oriented scheduling queue, which (voluntarily) serializes I/O intensive applications.

LMT records the bytes written and bytes read for each server every five seconds. That data shows the I/O resulting from the IOR jobs and anything else running at the same time. In order to calculate the average behavior we "warp" (artificially lengthen or shorten) the sequence of LMT observations for each job so that they fit the same length-scale – chosen as the median job run length. A standard linear regression on that data set provides the calculated average behavior.

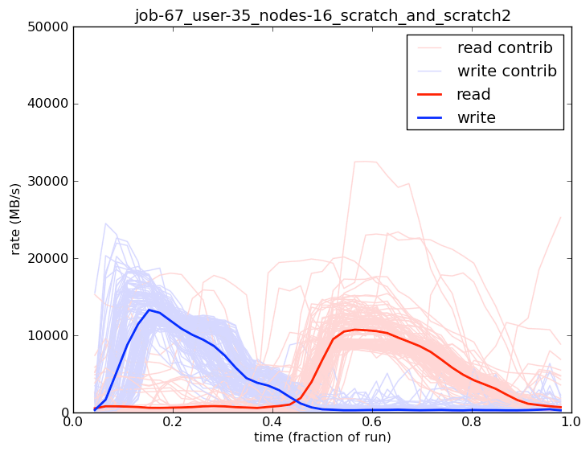


Figure 3.

Figure 3 displays the result of carrying out this analysis. The *x-axis* is the artificial time scale – arbitrarily set to 0 to 1 – to which each series of observations is warped. The *y-axis* gives the aggregate data rate (blues for writes and red for reads) of the application over the course of the idealized run. The single dark line of each color is the calculated average behavior of the application. Shown in a lighter shade is the collection of 175 separate contributing runs as they appear after being warped. Most of the contributing runs follow the average behavior closely, and demonstrate that the IOR test was running without much interference. A few traces depart wildly from the average and it is those runs that were in contention for I/O resources. Once we calculate the idealized average behavior all of the applications with significant amount of I/O, those idealizations become initial estimates for the coefficients in a big matrix implementing the generalized linear regression. For a file system that does not have a lot of I/O contention the initial estimates will be close to their final values and the computation will converge quickly. In other cases the computation may take significant resources. The end result is a quantified, job-by-job measure for the impact of the application on the I/O system from which we recover the utilization U and therefore the balance B.

CONCLUSIONS

The HPC community has developed a set of heuristics to guide the design of HPC systems so that the I/O capability is matched to the users' needs. In one case where the heuristic was applied in an effort to make a system *I/O-friendly*, a comprehensive characterization of balance using our metric showed that the system deployment was successful. Our measure for balance, $B = 2.5$, says that the system is cost effective for an even heavier I/O load than was observed.

The proposed job-log-and-LMT analysis extends the applicability of our metric to cases where direct observation of the utilization U is impossible or impractical. That characterization can be combined with system cost details to establish a rigorous evaluation of the balance of the system.

It is always difficult to argue that past performance is guide to future behavior. When planning for a new HPC system the application behavior produced in this analysis must be combined with theoretical considerations for how the new system might behave differently. Nevertheless, the application characterizations and the underlying model that produced them are a valuable starting point to be used during the procurement of new systems.

Once deployed, a new system needs data acquisition systems like IPM, Darchan [6], and LMT in order to evaluate the system balance actually achieved.

ACKNOWLEDGEMENTS

We would like to thank Dr. Anthony Gamst of UC San Diego for discussion, explanations, and guidance of the generalized linear regression techniques alluded to in this paper.

REFERENCES

1. NERSC Storage Policies, produced by the Storage Policies Working Group, 2010.
2. J. Shoopman, LLNL internal study on memory capacity to archive data generated 2008-2009.
3. D. Skinner, *Integrated Performance Monitoring: A Portable Profiling Infrastructure for Parallel Applications.* Proc. ISC 2005, International Supercomputing Conference, Heidelberg, Germany, 2005
4. A. Uselton, *Deploying Server-side File System Monitoring at NERSC,* Cray User Group Conference, Atlanta, GA, 2009
5. H. Shan, K. Antypas, J. Shalf, *Characterizing and Predicting the I/O Performance of HPC Applications Using a Parameterized Synthetic Benchmark.* Proc. SuperComputing 2008, Austin, TX, 2008
6. P. Carns, R. Latham, R. Ross, K. Iskara, S. Lang, K. Riley, *24/7 characterization of petascale I/O workloads*. Proc. Of 2009 Workshop on Interfaces and Architectures for Scientific Data Storage, Sep. 2009.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper - Business Breakout

Kim Cupps
Lawrence Livermore National Laboratory
cupps2@llnl.gov

ABSTRACT / SUMMARY

This "Business of File Systems and Archives" position paper will describe several LLNL best practices that help formulate and optimize the cost/benefit analysis all center's face in their quest to provide an ongoing, exceptional computing environment within a finite budget.

INTRODUCTION

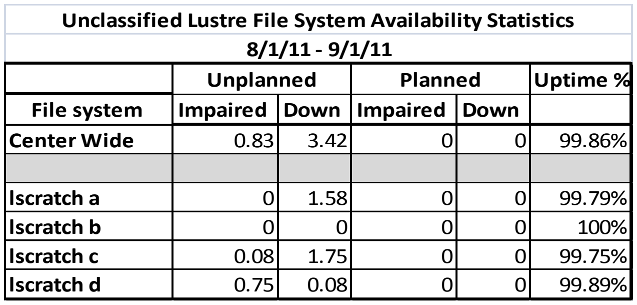
LLNL's primary computing complex serves approximately 2800 users with access to 1.7 peak PetaFLOPs of compute, 14PB and 300GB/s of parallel file system capacity and bandwidth and 42 PB of archival storage. LLNL seeks to optimize the user experience, providing a long-lived, highly productive environment for our customers, while staying within our budget. While cost/benefit is easy to say, it's a complicated balance of usability, availability, flexibility of administration, longevity, productivity and many other factors that form the basis of our spending decisions for file systems and archives. There are several practices we use to help us make decisions on what and how much to buy, what improvements must be made and which software to develop ourselves and which to buy off the shelf. First, we place a high value on gathering direct user input via a variety of mechanisms including user meetings, surveys and customer interviews. Another best practice is to monitor and measure use of resources and plan buys "just in time" (JIT). Strong partnerships with vendors as well as hedges against the trap of becoming beholden to a single vendor or technology for file systems or archive is another best practice. Lastly, planning is crucial to any coherent business strategy. LLNL formalizes the plan for file system and archive resources by producing the "I/O Blueprint", a procurement and effort planning prioritization document.

Gathering User Feedback

Making sound business decisions requires a good understanding of the current state of affairs. It's quite easy to live in a world isolated from those who use the file systems and archives every day, just as it is common for users to work around issues and inconveniences rather than report them. We have the typical trouble ticket system user surveys to help us understand issues; this is feedback we receive on a daily basis. In addition, LLNL holds quarterly user meetings that include a user talk as well as a set of talks on relevant center activities. These meetings include a general feedback session. Most important, LLNL rotates through "Science Team Interviews" so that we meet with teams every two years or so to elicit actionable feedback. A team of center personnel representing management, platform, file system, archive and user services personnel goes out to the customer work area and asks pointed questions about the compute environment. In general, we find that problem areas spring out of discussion and are not the sorts of issues that people call and report, often they don't even write down the issue in advance of the meeting. For example, the development of HTAR, a multi-threaded file packaging and transfer mechanism from the local file system to the HPSS archive, was the direct result of complaints received from users regarding slow transfers of small files to HPSS. The development of Lorenz, our user dashboard that shows file system usage, NFS quotas and many other customizable fields was also the result of strong user collaboration and input. All of these user feedback mechanisms serve to inform the center about where our customers feel we have the most room for improvement – this is critical to our planning.

Measuring and Metrics, JIT

Another way we identify areas for improvement is by collecting data on various aspects of the production environment. We gather data on everything from component failure rates, to sizes of files stored in the file system, to file system specific and center-wide uptime percentages for both classified and unclassified file systems and archives.



Unclassified Lustre File System Availability Statistics

8/1/11 - 9/1/11

	Unplanned		Planned		Uptime %
File system	Impaired	Down	Impaired	Down	
Center Wide	0.83	3.42	0	0	99.86%
lscratch a	0	1.58	0	0	99.79%
lscratch b	0	0	0	0	100%
lscratch c	0.08	1.75	0	0	99.75%
lscratch d	0.75	0.08	0	0	99.89%

Tracking isn't limited to analysis of failures and availability, it's also crucial to our "just in time" purchase strategy for archive media and tape drives. Cartridges are used up by both a steady stream of new data being written to tape as well as an ongoing repack from soon-to-be-retired media of about 500 cartridges a month. Careful tracking of cartridges insures that tape buys are done on-time, but not so far in advance that the tape is never used. Just-in-time has been shown to be the most cost efficient way to purchase consumables, and with tape densities doubling (recently quintupling) every year and a half, the value of this best practice is clear.

Partnerships Coupled with In-House Software Expertise

Strong vendor partnerships are crucial to successful operation of a center. Changing vendors incurs added costs such as retraining staff, forming new relationships and learning new support processes. However, becoming a strictly one vendor operation significantly increases risk. These risks include the company going out of business, dropping support for the product line or unreasonably raising prices. A best practice at LLNL that reduces the inherent risk of the strong vendor partnership, is a staff of in-house software developers forboth open source projects (Lustre, SLURM, RHEL) and joint development contracts (HPSS). The software developers provide key value by:

1) Solving production problems immediately (increasing system uptime and thereby user productivity);
2) Providing a strong voice for DOE HPC requirements
3) Providing a mechanism for the center to remain technically competent and engaged in leading edge technology

Other important components of strong vendor partnerships include membership on vendor customer advisory boards, leadership of product user groups and regular attendance at vendor executive level roadmap briefings.

Advanced Technology and Testbeds

The Hyperion testbed at LLNL includes an 1152 node QDR IB interconnected commodity Linux cluster with two SANs (GE, IB) connected to multiple vendor storage subsystems. The testbed serves multiple purposes. It is a partnership that allows vendor partners to test their software and hardware at scale. It is a platform that allows LLNL to investigate interesting technologies

(NAND Flash, tiered storage, NFS accelerators…) as they become available. WhamCloud performs Lustre testing at scale. Mellanox tests new cards and drivers. DDN and Netapp test new controller technologies and LLNL investigates all of this new technology before it comes to market.

The Lustre testing at scale on Hyperion, and LLNL's participation with others in the concept of creating "Lustre Centers of Excellence" is an excellent example of how investment in strong vendor partnerships and testbeds can significantly impact the quality of a product.

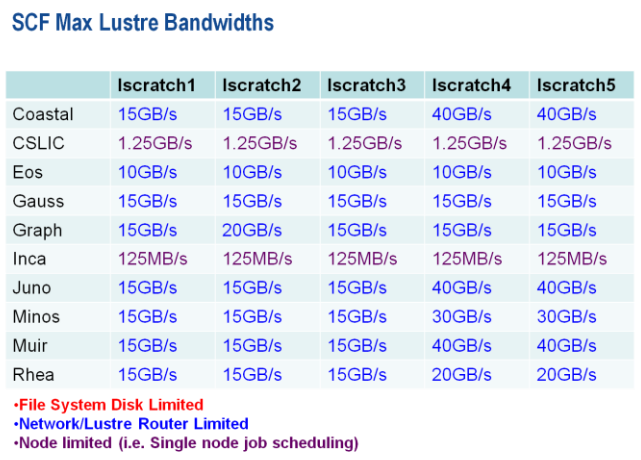
Planning

LLNL produces a yearly planning document called the I/O Blueprint. The goal of the Blueprint is to achieve a balanced infrastructure to support the Center's compute platforms. The Blueprint documents planned purchases in global parallel file systems, NAS, visualization, network and archive areas. It also discusses area specific Center issues and plans for remediation.

The FY05 Blueprint is the document that called for converting from dedicated filesystems to a site-wide global parallel file system. During the era of local file systems, each platform purchase required that dedicated file system hardware be bought for use solely by that platform. Without global file systems, a platform was only able to leverage the speed and capacity that it came with and data needed to be moved or copied to each platform when required. Today, global file systems allow new platforms to leverage existing disk resources and allow existing platforms to take advantage of global resources added over time. As a result we are able to enhance file system resource utilization, eliminate the copying of data, ensure that file system hardware is best-of-breed rather than that available from a particular platform vendor, and focus on center-wide I/O requirements rather than that of individual machines. In short, there is a clear cost/benefit win.

Calculating bandwidth and capacity requirements for archive, file system and networks is not an exact science, and we have used different "rules of thumb" over time to plan purchases. For example, directly copied from the FY08 I/O Blueprint: *"The rule of thumb used in the past for capability platforms was that the file system should provide between 100MB/s and 1GB/s of bandwidth for every TeraFLOP. Dawn file system bandwidth requirements are projected to be 200MB/s per TeraFLIN and Sequoia is projected to be 100MB/s per TeraFLIN leading to 100GB/s and 500GB/s estimates for delivered SWGFS bandwidths for these machines."*

From our FY11 Blueprint: *"For many years now, bandwidth has been the basis for our file system procurements. We believe that our bandwidth requirement is a function of platform memory. Typically our ratio of file system bandwidth to platform memory (GB/s per TB of memory) has varied from 0.6 to 0.8. Currently that ratio is at 0.5GB/s/TB in the OCF and 0.6GB/s/TB in the SCF. Note that the Sequoia file system is currently 0.4GB/s/TB."* As they should, requirements definition methodologies have changed as architectures evolve and lessons are learned.

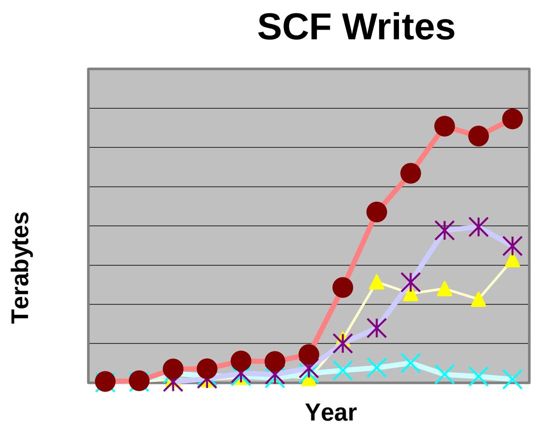

SCF Max Lustre Bandwidths

	lscratch1	lscratch2	lscratch3	lscratch4	lscratch5
Coastal	15GB/s	15GB/s	15GB/s	40GB/s	40GB/s
CSLIC	1.25GB/s	1.25GB/s	1.25GB/s	1.25GB/s	1.25GB/s
Eos	10GB/s	10GB/s	10GB/s	10GB/s	10GB/s
Gauss	15GB/s	15GB/s	15GB/s	15GB/s	15GB/s
Graph	15GB/s	20GB/s	15GB/s	15GB/s	15GB/s
Inca	125MB/s	125MB/s	125MB/s	125MB/s	125MB/s
Juno	15GB/s	15GB/s	15GB/s	40GB/s	40GB/s
Minos	15GB/s	15GB/s	15GB/s	30GB/s	30GB/s
Muir	15GB/s	15GB/s	15GB/s	40GB/s	40GB/s
Rhea	15GB/s	15GB/s	15GB/s	20GB/s	20GB/s

•File System Disk Limited
•Network/Lustre Router Limited
•Node limited (i.e. Single node job scheduling)

The Blueprint is also the document where LLNL outlined an initial plan to address exponential archive growth and associated unmanageable out-year costs. The LLNL Archive Quota implementation was planned as a first mitigation. Archive advisory quotas were implemented in

December of 2010. Initial talks with users were held beginning in August of 2009. Archive growth rates have slowed. We conjecture that this slow down is due to a number of factors, not just the quota implementation. First, simply communicating the cost of storing a particular user's data resulted in that user deleting over 1PB of data in the archive. Raising awareness of costs is a best practice that we have used with very good results. At LLNL, nothing is archived automatically; all transfers are initiated by users. Activities that increase storage to the archive include aggressive global parallel file system purge policies, file system retirements and planned file system down times. A reduction in one causes a reduction in the other. Finally, the biggest factor in archive growth is platform memory capacity. As new large platforms are added, archive growth increases. We expect substantial archive growth with Sequoia and we expect the archive advisory quota implementation to help contain the rate of growth over time. While we expect the Quota implementation to help, it's too early to claim it as a best practice.



CONCLUSIONS

Providing a balanced infrastructure to optimize user productivity, while minimizing costs, requires attention and focus in a number of areas. The cycle of events includes formalized planning, which is informed by regular collection of data and metrics, user feedback, advanced technology investigations and testbed evaluations. Strong vendor partnerships and in-house software expertise are key enablers to quickly moving forward and providing the best environment possible.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

David Cowley
Pacific Northwest National Laboratory
david.cowley@pnnl.gov

ABSTRACT / SUMMARY

The EMSL facility, located at Pacific Northwest National Laboratory (PNNL), operates terascale HPC and petascale storage systems to support experimental and computational researchers in molecular sciences. This position paper addresses the Workshop's Business of Storage Systems track and describes EMSL's approach to operating file systems and data archives.

INTRODUCTION

The Environmental Molecular Science Laboratory (EMSL) is a scientific user facility located at PNNL. EMSL houses PNNL's largest concentration of high performance computing systems and data storage systems. While other organizations within the Laboratory are working on obtaining their own significant HPC and data storage resources, the center of mass has not shifted yet. We will be careful in this document to distinguish between PNNL and other sub-organizations within PNNL, including EMSL.

EMSL operates a suite of cutting-edge scientific instruments, capable of generating terabytes of data per week. EMSL has operated HPC systems ranked in the top 20 of the Top500 list since 2003, in addition to a multi-petabyte archive for scientific and computational data. EMSL has been working since 2010 on a scientific data and metadata management system known as MyEMSL.

GENERAL APPROACH TO STORAGE SYSTEMS

EMSL HPC systems have had more types of filesystems than at most HPC sites. Each is intended to meet different levels of capacity, performance, and accessibility. So far each of EMSL's HPC systems has been procured with its own filesystems of 3 types:

Filesystem Type	Capacity	Nominal Bandwidth
Global Home	20 TB	1 GByte/sec
Global Scratch	277 TiB	30 GiByte/sec
Node Scratch	350 GiB/node 808 TiB Aggregate	400 MiByte/sec/node 924 GiByte/sec Aggregate

The global home filesystem is available to all nodes in the HPC cluster. its capacity is determined loosely by a "not too big to be backed up" rule of thumb, its performance is determined loosely by a "'cd' and 'ls' commands have to not be slow" rule of thumb.

The global scratch filesystem is a parallel filesystem both larger and higher performance than the home filesystem. It is available to all nodes in the HPC cluster, and its performance and capacity requirements have been derived from a formula based on the theoretical peak performance of the system. EMSL does not have a requirement to checkpoint whole-system jobs as some other sites do, so this eases some of the requirements on this filesystem.

The node scratch filesystems provide high disk bandwidth per Flop to each node. For these filesystems, performance in terms of write bandwidth and Ops/second are again derived from theoretical peak performance on the compute node. Capacity has been a side effect of the need to provision enough disk spindles to meet the required performance. This may change as magnetic disk and solid-state disk technologies evolve. While providing a scratch filesystem on each compute node does involve considerable cost and added maintenance, the aggregate performance has been more scalable and better in absolute terms than shared parallel filesystems. EMSL has found this to be a differentiating and enabling capability, and will carefully consider it in its upcoming system procurements.

EMSL is planning to move to a "two systems" approach where rather than procuring one large system every 3 to 4 years, we will procure smaller systems every two years and overlap their lifecycles. We will switch to having the home filesystem shared between compute clusters. We expect that each cluster will have its own high performance parallel global scratch filesystem. We will consider critically whether new systems require node scratch filesystems.

MANAGING ARCHIVE GROWTH

EMSL's growth in archive capacity is driven by two factors, the output of scientific instruments and the output of its HPC systems. In effect, the scientific instruments are computers themselves, as is the HPC system, so Moore’s law drives data growth rates in both cases. Fortunately magnetic media growth rates (sometimes cited in "Kryder's Law") are on a similar trajectory so storage systems likewise exhibit the behavior of offering twice the capacity for roughly the same cost year over year. This behavior is expected to continue through 2020 [1,2].

This allows us to provide space for exponential data growth as long as a relatively consistent storage budget is available year-to-year. Successive generations of storage have so far had the sheer capacity to swallow up data from earlier generations of technology, provided there is a bridge between the technologies. Ensuring that there is such a bridge between generations is feasible provided there is sufficient planning and investment both in time and dollars to execute it.

Exponential growth rates *are* sustainable with proper planning and funding, but this only provides for storage *space*. By itself, this does not address the problems of managing, understanding, or using the accumulation of data. To that end, EMSL is investing in creating a new scientific data and metadata system known internally as MyEMSL. MyEMSL is addressed in the PNNL position paper for the Usability of Storage Systems track.

SOFTWARE FOR FILE SYSTEMS AND ARCHIVES

EMSL uses the software technologies that best fit its needs and budget, whether open source or proprietary. As much as possible, we wrap proprietary solutions so that they play well in an open-source environment. We were an early adopter of the Lustre filesystem, having used it since the implementation of our MPP2 system in 2003. We have built low-cost filesystems out of commodity hardware up to 1.2 petabytes (the "NWfs" storage system in 2008), and PNNL is building a similar institutional Lustre storage system that will have a 4-petabyte capacity by the end of fiscal year 2011. In 2008, EMSL identified a need to implement a hierarchical storage system, and in 2009 retired NWfs in favor of a new HPSS system.

HPSS provides the right mix of capacity, expandability, and scalable performance for EMSL's needs. The EMSL HPSS system provides archive storage capacity, and we have implemented open source filesystem-like interfaces to it, in addition to the traditional native HPSS interfaces.

EMSL and the rest of PNNL continue to make use of Lustre and will continue to do so until it is clearly dead or orphaned. At this point, PNNL has enough experience and expertise to not require Lustre support. Even if advanced and long-promised features (e.g. multi-way clustered metadata) are never delivered, Lustre's cost, performance, scalability, and "good enough for us" reliability meet our needs very well.

The difficult to control costs are in additional work scope, i.e. supporting more systems or more users without attendant increases in budgets. Inflation alone causes increased labor costs over time, creating difficulty in operating with flat or declining budgets. Additional work scope compounds this problem if not very carefully managed.

HARDWARE FOR FILE SYSTEMS AND ARCHIVES

Being at the upper-mid range of HPC in terms of system sizes and performance, EMSL is not using and does not expect to use custom hardware in the foreseeable future. We take the best advantage we can of common off the shelf hardware and the economies of scale that come with it.

EMSL does expect to continue to take advantage of commodity storage technologies for the foreseeable future, mostly in conjunction with the Lustre filesystem. Selected high-value storage systems may be constructed of enterprise-grade storage for serviceability features. While we may apply creative engineering approaches to commodity or enterprise-grade building blocks, we do not foresee significant use of custom storage hardware.

I/O capacity and bandwidth requirements for filesystems on EMSL HPC systems are established as a function of peak performance ratings. We have not carefully specified metadata operation or operations/second requirements on our filesystems, though we have re-engineered metadata servers to improve performance when there is a need to do so. MTTI requirements have not been rigorously specified either, though we do specify that common failures (e.g. single disk failure on a node) must not interrupt computation or I/O. During technical review, we assess whether the I/O system is robust enough to remain serviceable with good maintenance procedures. It has been said, "we don't need five nines, we just need two or three!"

Most of the barriers we see to adoption of commodity storage have to do either with low performance or lack of Reliability, Availability and Serviceability (RAS) features. In our experience, neither of these has presented insurmountable difficulties. The engineering approaches and software tools we apply allow performance to be scaled linearly (or nearly so) by adding more components. The essential RAS features we need are typically available in mid-grade commodity or enterprise hardware. At the other end of the spectrum, many higher end RAS features such as active-active failover/failback prove to cause as much downtime as they are advertised to prevent!

SYSTEM EVOLUTION

EMSL plans a three to four year lifetime for its HPC systems, and has recently decided to switch from operating one large HPC system to two smaller systems with overlapping lifecycles. With this change, we will pull the persistent "home" filesystem out of the cluster and place it where it can be shared between systems and provide continuity between HPC systems as they age out and are replaced.

EMSL procured a new HPSS storage system for archive purposes in 2009, and plans to operate it through at least 2017, with planned lifecycle replacements and technology refreshes for the storage (disk and tape) components.

CONCLUSIONS

EMSL employs multiple tiers of data storage systems with different capacity and performance characteristics to satisfy various needs. Storage system capacities are planned based upon projected output from the facility's scientific instruments and from HPC system performance. All storage systems have a planned lifecycle with expansions, technology refreshes, and retirement as appropriate. EMSL generally uses commodity or enterprise-class components as building blocks, in concert with a mixture of open source and proprietary software.

REFERENCES

1. Kryder, H, Kim, C. *After Hard Drives – What Comes Next?.* IEEE Transactions on Magnetics, Vol. 45, No. 10, October 2009 *http://www.dssc.ece.cmu.edu/research/pdfs/After_Hard_Drives.pdf* .
2. Zyga, L. *What Comes After hard Drives?* http://www.physorg.com/news175505861.html .

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper: Reliability and Availability

John Gebhardt
Lockheed-Martin/U. S. Air Force Research Laboratory
DoD High performance Computing Modernization Program
john.gebhardt@wpafb.af.mil

Thomas Kendall
U. S. Army Research Laboratory
DoD High performance Computing Modernization Program
thomas.m.kendall4.civ@mail.mil

Cray J. Henry
DoD High performance Computing Modernization Program
cray@hpcmo.hpc.mil

ABSTRACT / SUMMARY

The DoD High Performance Computing Modernization Program (HPCMP) has implemented a multilayered storage approach to cost effectively meet the storage needs of a diverse customer base. Users' can wait in the batch queue indefinitely (but typically start within seventy-two hours) and then can run for up to fourteen days (or longer with special arrangements). To maximize systems availability, several layers of storage and storage use policy are implemented.

INTRODUCTION

The HPCMP has layered several storage and file systems within the environment. Each system has specific reliability and availability characteristics and use policy driven by system availability requirements.

This paper discusses the reliability and availability of the following types of storage constructs within the HPCMP:

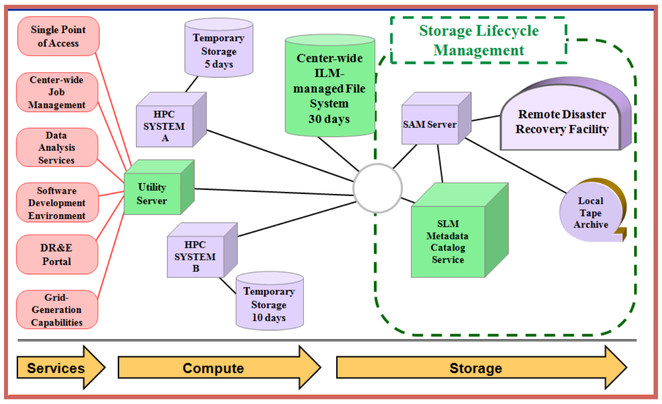
- HPC scratch space file system
- HPC Home and Applications file system
- Root services file system
- Center Wide File System (CWFS)
- Lifecycle Management System
    - Archive system
    - Tape storage

Data Center Facilities

The Department of Defense Supercomputing Resource Centers (DSRC) operates twenty-four hours a day, seven days a week. Each of the DSRCs utilize different combinations of UPS, redundant commercial power feeds and diesel backup power generation capabilities to allow operations to continue through minor power fluctuations and allow for graceful equipment shut for prolonged power outages. In the event of a prolonged power or cooling failure, procedures are activated to shutdown the systems, which in turn quiesces the storage. This approach nearly eliminates unplanned, abrupt outages.

. File System Overview

Each DSRC hosts a CWFS which supports the lifecycle management system, the utility server and each HPC system. Each HPC system typically includes a combination of RAID storage devices that are logically decomposed into three file systems -- Scratch space, Home and Applications and Root services. The utility server is similarly configured and the lifecycle management system includes multiple data stores, archival servers and agents described later.


Single Point of Access
Center-wide Job Management
Data Analysis Services
Software Development Environment
DR&E Portal
Grid-Generation Capabilities
Temporary Storage 5 days
HPC SYSTEM A
Utility Server
HPC SYSTEM B
Temporary Storage 10 days
Center-wide ILM-managed File System 30 days
Storage Lifecycle Management
SAM Server
Remote Disaster Recovery Facility
SLM Metadata Catalog Service
Local Tape Archive
Services
Compute
Storage

Center Wide File System

The HPCMP has recently deployed Center Wide File Systems (CWFS) at each DSRC. The purpose of the CWFS is to provide users with a fast central storage capability that can be easily accessed by all major HPC systems and servers within the center. It is intended to serve as the "near-HPC" intermediate storage between scratch file system on each HPC system and the long-term archive. It has been sized to providing a minimum of thirty days of quick intermediate storage. Through CWFS users can move their entire data sets among the scratch file systems and the archive file system. They can perform pre and post processing on their data conveniently avoiding the slower access times associated with archived data. This approach affords users the time necessary to make more thoughtful decisions on what data, for example after a large run, really needs to be archived and what can be deleted.

The CWFS is not backed up. It does contain all the redundancy features of HPC scratch with the addition of check sums on read from storage to host.

HPC Scratch Space File Systems

The HPC systems are in high demand. The data sets used to set up runs and the data resulting from runs is very large and very transitory. In order to assure there is sufficient scratch space to stage the next job, HPCMP policy allows for user data to exist on HPC scratch storage for 10 days. Within ten days after data creation, users must move their data to the center wide file system or archive. After ten days the data it is subject to removal to make space available for the future jobs.

Like the CWFS, the HPC scratch space is typically not backed up due primarily to the transient nature of the data and the amount of data.

HPC scratch storage systems are normally procured with the HPC system. The HPC vendors propose the file systems and storage architecture as components within an overall HPC system. The HPCMP request for quotations (RFQ) for HPC systems states that the storage system must be architected to be resilient and robust; highly reliable components are to be utilized.

The HPCMP's RFQ defines the minimum aggregate data transfer rates between the compute nodes and the disk subsystem are based on specified ratio values for total system memory bandwidth (GB/s) to 1000 times the disk subsystem I/O bandwidth (GB/s). These ratios are 2.07, 1.68, and 1.34 for read, write, and full-duplex respectively. For example, a 200 node system with 50 GB/s of memory bandwidth per node would have total system memory bandwidth of 10000 GB/s. To meet the minimum full-duplex requirement, the system would require an I/O bandwidth of 7.46 GB/s (i.e. (200*50 GB/s)/(1000*1.34)). The minimum formatted usable disk storage size must be at least 40GB per processor core.

HPC vendors must also commit to monthly interrupt counts and overall systems availability (> 97%). This encourages the vendors to offer reliable storage systems due to the penalties imposed for not meeting the system availability commitments.

The files systems end up on RAID protected storage that is either RAID 5 or RAID 6. RAID 6 is becoming more common place which is due to the increasing scratch space sizes which are architected with increasingly larger and lower costs SATA disk drives. These large drives take much longer to rebuild leaving the RAID set vulnerable to another drive failure. Vendors architect the storage with redundant paths from the hosts and multiple controllers. Metadata redundancy and availability is expected.

In order to maximize performance HPC scratch storage does not have end-to-end protection mechanisms or check summing.

Downtime for preventative maintenance may be executed at the recommendation of the HPC vendor. Typically, these downtimes are to update the HPC operating environment and not necessarily for the storage systems.

Support for the systems and storage is twenty four hours a day, seven days a week, four hour onsite support for hardware problems.

HPC Home and Applications

Home and application file system on the HPCMP HPC systems are considered more permanent then the HPC scratch file system. These file systems typically are hosted on the same storage as the HPC scratch space and utilize the same file system software.

With only minor exception, home and application file systems protected in the same manner as the HPC file system. Home and application file system are backed up on a daily basis.

Root File Systems

Root drives on major infrastructure servers and key elements in an overall HPC system (login nodes, admin nodes, etc) predominately are architected with multiple disk drives that are protected with RAID 1 or RAID 10. The costs for additional disk drives vs. performance make this a very worthwhile architecture decision.

Compute nodes within an HPC system that are architected with disk drives do not include redundancy. Since the disk drive images on compute nodes are easily reproducible, redundant drives in a RAID configuration are not employed.

Archive File Systems

Arguably, the tape archive systems are one of the HPCMP's most important systems which require the highest level of availability. Prior to integrating the CWFS, and the short time to live for data on HPC scratch the archive had to always be available to the user.

In addition, a user can have their HPC batch jobs in the work load management system queues in some cases up to fourteen days prior to the job running on the HPC system. Jobs that require input data from the archive system would not necessarily want to move their data immediately with the short time to live of the data in HPC scratch.

The DSRCs have architected redundant servers, redundant server component, redundant SAN switches, tape drives, very high speed RAID 5 or RAID 6 disk caches and metadata devices for the archive systems.

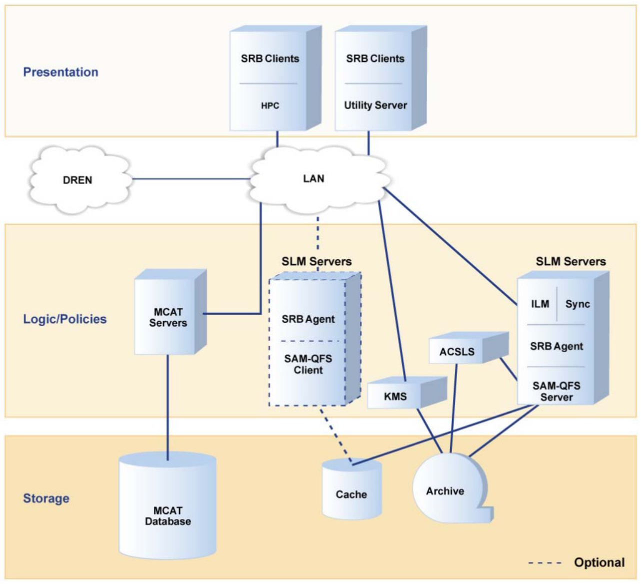
To maintain this high availability, the archive systems have been maintained at twenty four hours a day, seven days a week, with four hour response time.

Implementation of an active-active redundant archive server solution remains a priority.

Tape Archives

The HPCMP currently provides the user by default, two copies of their data on tape. One copy is at the DSRC and the other copy is sent via network to an archive system in another facility and then copied to tape. Archive file system metadata is backed-up daily and stored locally as well as remotely.

Storage Lifecycle Management



The HPCMP is currently implementing storage life cycle management (SLM). SLM tightly couples the current archive systems with an Integrated Lifecycle Management (ILM) management tool. The ILM is an Oracle Real Application Cluster (RAC) environment that will contain the file metadata from the archive as well as user applied metadata such as whether or not to make a disaster recovery copy, when the data can be deleted, what project(s) that data belongs to, etc.

The ILM is architected with multiple Oracle servers via Oracle RAC for redundancy and scalability. Additional reliability features incorporated in the design include redundant server components, a performance disk subsystem for the Oracle databases, utilizing multiple fiber channel paths per server, RAID 5 and RAID 10 volumes, redundant controllers, and redundant network interfaces connected to redundant switches.

CONCLUSIONS

In order to maximize HPC cycles for researchers, the HPCMP will continue to employ redundancy and other availability measures where practical to maintain availability of the systems, file systems and storage. The HPCMP would like to see the vendor community continue to develop the capabilities for end-to-end data protection (e.g T10DIFF) to ensure bit error rates are extremely low and that bit errors are identified and corrected. As storage space on disk drives continues to grow, new RAID schemes to decrease rebuild times and maintain file system performance are desired and would be implemented.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives San Francisco, CA September 26-27, 2011 Position Paper

M'hamed Jebbanema
Los Alamos National Labs
mjebb@lanl.gov

ABSTRACT / SUMMARY

This paper will discuss a strategic and automated scripted solution to ensure High Performance Storage System (HPSS) metadata integrity, availability and recoverability in the event of a disaster.

INTRODUCTION

An essential component of High Performance Storage System (HPSS) is the metadata and tools to manage and retrieve the metadata. Metadata can be described as the DNA of the storage system as metadata defines the elements of the transactional data and how they work together. Any loss of metadata proves to be disastrous to data integrity**.**

In addition to implementation of resilient and fault tolerance hardware and software best practices, we must guarantee the high availability and recoverability of metadata.

Since HPSS uses IBM DB2 as its metadata management system, manual and conventional DB2 standard backups and lack of logs archival monitoring tools dramatically increase administrator workload and probability of data loss.

A Perl language based comprehensive DB2RS (DB2 Recovery Solution) delivers a robust, configurable, and customizable recovery management with minimal efforts.

The Scheduler, Backup, Verifier and Checker(s) services work intrinsically in a holistic approach by using SQLite database as a central repository for all DB2RS activities.

This presentation will cover the design, and integration of DB2RS to ensure metadata integrity and recoverability when disaster occurs.

Paper Content

In order to achieve transaction integrity and zero or little data loss , DB2RS makes automated scheduled local backups including logs to different media combined with an-offsite hosting similar backups in which to restore from in the event of a disaster.

Goal:

Develop a metadata backup/recovery solution that is simple, customizable, robust, and easy to maintain**.**

Objective:

Provides recoverable copy of databases.
Metadata can be recovered to any Point In Time (PIT).
Guarantees transactional consistency.

Purpose:

Develop scripts that would leverage DB2 integrated utilities and tools to automate all functions ensuring metadata recoverability.

1.Design

1.1 Logging

Proper DB2 log file configuration and management is an important key to data stewardship and operational availability.

All database changes (inserts, updates, or deletes) are recorded in the DB2 transaction logs. Transaction logs are primarily used for crash recovery and to restore a system after a failure.

DB2 does have the ability to perform dual logging on different volumes as well as different media thereby increasing redundancy for both active logs and archive logs.

We Configure DB2 log sets by implementing MIRRORLOGPATH and dual log archives where each active & archive log set on independent LUN that uses separate physical disks and different type media (TSM) (Figure 1&2).

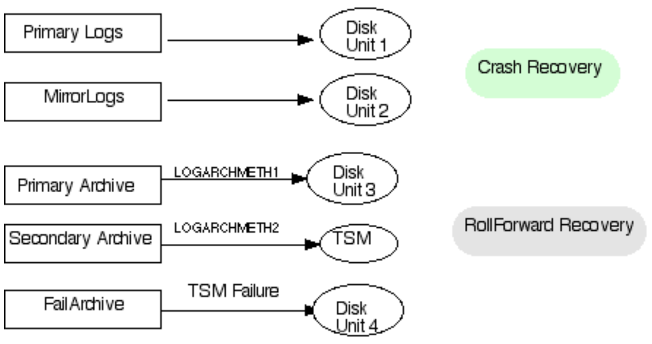
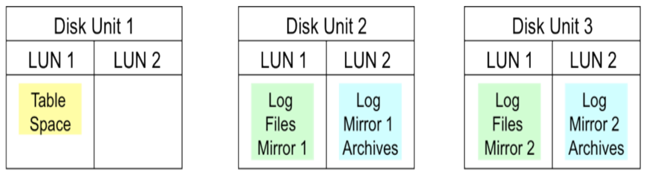


Figure 1. Logging Scheme



- Disaster recovery: Excellent. Short of fire, flood, etc., DB2 can always be recovered.
- Operational availability: Excellent, as good as you can get without going to high availability DB2s. Only interruption would be loss of Disk Unit 1 or simultaneous loss of Disk Unit 2 & 3

Figure 2: Metadata Hardware Fault Tolerance Implementation

1.2 Backup service

The scope of this design goes beyond the process of backing up objects to disks or tape but rather encompasses several functions that ensure the validity and integrity of the backups.

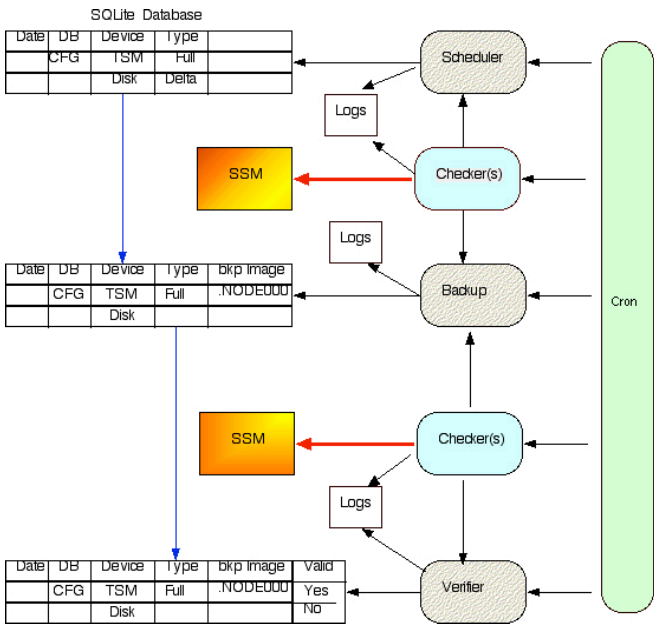


Figure 3: Overview of DB2RS

Perl Language was an evident choice to automate these tasks in order to take advantage of our locally developed Perl modules and for backward compatibility reasons.

Scripts were structured based on type of service or function and SQLite database (Example 1), an open source, self-contained, embeddable, zero-configuration was chosen as a central repository for all services activities.

Four types of services (Figure 3) were automated and can be run either via cron (Example 2) or manually.

Scheduler : Schedules backup in SQLite database.
Backup : Checks SQLite for scheduled backups and perform backup service.
Verifier : Validate the integrity of backups.
Checker & High Level Checker : perform diagnostics, sanity checks, monitoring and error reporting

Each service has multiple parameters entries in configuration (Example 3) file subject to customization based on disaster recovery requirements.

To prevent invalid data and allow synchronization among all services, applications state and locks were included in the initial design.

All services activities are captured in a centralized log and a man page was integrated into the code for quick reference. (Example 4)
Other useful utilities were coded and added to the mix to ensure completeness and automation of DB2RS.

Example 1: SQLite database service activities

```
20110816000601    CFG  TSM   FULL      1
20110816041003  Verified
20110817000306   SUBSYS1   TSM   INCREMENTAL  1
20110817051002  Verified
20110817000609  CFG      TSM   FULL        1
20110817045902  Verified
20110818000301   SUBSYS1   DISK  FULL        1
20110818045903  Verified
20110818000606  CFG      DISK  FULL        1
20110818041032  Verified
20110819000301   SUBSYS1   DISK  INCREMENTAL  1
20110819045902  Verified
```

Example 2: Services scheduled via cron

```
06 0 * * 1,2,3 schedulme -cron -db cfg -tsm -full -sessions 1
10,59 4,5 * * * bkp -cron > /dev/null 2>&1
02 6,7 * * * bkpv -cron > /dev/null 2>&1
0,41 * * * *  bkplogtrim -cron > /dev/null 2>&1
09 12 * * 4   checker -cron > /dev/null 2>&1
09 12 * * 1-3,5 checkerhl -cron -disk > /dev/null 2>&1
23 * * * *  ensure_archlogs -cron > /dev/null 2>&1
```

Example 3: DB2RS configuration file

```
[timemachine]
# bkp: should we make another bkp if one full bkp already exists
within bkpDiffHrs (integer hrs);
bkpDiffHrs = 20
#bkpv: looks back in SQLite for unverified bkp images earlier
than diffWkBkpv;  24hrs
diffWkBkpv = 86400
#checkerhl & checker: each service must have run successfully in
the last (n) days; 3 day
ChkDysServSuc = 259200
#chekerhl: calculate timestamp before which combo bkps and logs
should exists; 1wk
ChkhlWks   = 604800
# checker: all services Service must have run within; 8 hrs
ChkHrsSerbkp  = 64800
ChkHrsSerbkpv  = 28800
ChkHrsSersched  = 28800
#checker: check bkp service progress run..(avoid runaway and
hangs)..service must not be running
# for more than ChkHrsRunbkp; 2 hrs
ChkHrsRunbkp = 7200
# checker: if no bkp within ChkHrsBkpSched after being
scheduled; 20 hrs
 ChkHrsBkpSched  = 64800
 # checker: stats Failover paths for existence of logs (any files)
 within the last ChkHrsFailover; 1 hr
 ChkHrsFailover  = 3600
 # ChkHrsFailover  = 0 # test only; make sure checks paths
 immediately..no delays
 # checker: row created in Sqlite for each db combo int he last
 ChkWksSqlite; 1wk
#ChkWksSqlite  = 604800
 ChkWksSqlite  = 3600
[constantvars]
DB2DIR = /opt/ibm/db2/path
[db2]
databases = cfg, etc
[db2:cfg]
images = /usr/db2/image/path
archlogs = //path/path/etc..
failarchlogs = /usr/db2/path/path/etc….
imagespct = 60
archlogspct = 60
tsmstartdate = 20110202
```

Example 4: Man page

```
ENSURE_ARCHLOGS(1)    User Contributed Perl
Documentation   ENSURE_ARCHLOGS(1)
NAME
       ensure_archlogs - Ensure that database has truncated
       logs recently
SYNOPSIS
        ensure_archlogs <options>
        Within root's or instance owner crontab...
        01 * * * * /lanl/hpss/path/db2rs/ensure_archlogs -cron
        On the command line as root or instance owner...
        # ensure_archlogs -force  Force a log archive right
          now
        # ensure_archlogs -force=2h  Archive if not performed
           in last 2 hours.
        # ensure_archlogs  Same, but use default intervals from
          the dbconfig
        # ensure_archlogs -disable=1h   Disable scripts in cron
          for 1 hour
        # ensure_archlogs -enable      Re-enable scripts in cron
        # ensure_archlogs -help        Show the synopsis for  this
          script
        # ensure_archlogs -man     Show the man page for this
          script
DESCRIPTION
       This should generally be run via cron.  It makes sure
that DB2 has archived a log within a certain amount of time
for  each HPSS database.  If it hasn’t it tells it to do that with
the "db2 archive log" command.  This limits the exposure of
```

Un-archived logs to a certain period of time while also allowing minimum impact on DB2. This should probably not run more frequently than one hour.

1.3 Checker(s) and error reporting service

Applies to Logs and backups.

1.3.1 Checker (Example 5)

1.3.1.1 Logs Checks:

Performs the following tasks:
Exclusive analysis utilizing log data mining list of events.
Disk & TSM logging failures detection.
FailArchive path check.
Immediate reporting and notifications to SSM.

1.3.1.2 Backup Checks:

Performs the following tasks:
Sanity Checks
Diagnostic Checks
Immediate reporting and notifications to SSM.

Example 5 : Check output

```
info: No scheduled backup found at this time!
info: checker: No DB2 Logs in Failover paths...good thing
info: checker: Found all 4 scheduled combination backups in Sqlite database..schedulme is working fine
info: checker: Last successfull bkp run at 20110817055901
info: checker: Last successfull bkpv Run at 20110817071029
info: checker: Last successfull schedulme run at 20110817000609
info: checker: bkp service has run within defined time (Hrs).
info: checker: bkpv service have run within defined time (Hrs).
info: checker: schedulme service have run within defined time(Hrs).
```

1.3.2 High-level Checker (Example 6)

Performs the following tasks:
Recent backup for each (database, device) pair completed successfully.
Recent TSM backup for each database must exist and verified
Ensure that TSM copies of all logs since the last verified TSM backup exists.
Immediate reporting and notifications to SSM.

Example 6: High Level Checker (Checkerhl) output

```
info: Found 56 logs for CFG DISK backup taken at 20110813041003 (2314 - 2369)
notice: Everything looks good for CFG DISK backup taken at 20110813041003
info: Found 99 logs for SUBSYS1 DISK backup taken at 20110811045903 (3116 - 3214)
notice: Everything looks good for SUBSYS1 DISK backup taken at 20110811045903
info: Found 96 logs for CFG TSM backup taken at 20110810041003 (2274 - 2369)
notice: Everything looks good for CFG TSM backup taken at 20110810041003
info: Found 146 logs for SUBSYS1 TSM backup taken at 20110808045903 (3069 - 3214)
notice: Everything looks good for SUBSYS1 TSM backup taken at 20110808045903
info: Found all 4 backup combinations
```

1.4 Error Reporting to SSM (HPSS interface):

Exit codes are called to generate sub-class of errors based on custom binary scheme for multiple error reporting.
To simplify error reporting and monitoring, three (3) categories were considered to match HPSS error reporting style.

Minor
Backup, verifier, or scheduler service did not run within defined time (Hrs).
Backup have exceeded estimated allowable time to successfully complete backup.

Check ASAP (considered critical)
Failure to schedule expected pair (db,device) within the last (days).
TSM backups and associated logs are not found.
One or more services failed in the last (n) days.

MAJOR
Scheduled Backups are behind schedule.
Backup service did not run.
Backup failed (n) times as specified in cron.
Backup could have completed successfully but took longer than expected/estimated.
Failover DB2 log paths contain logs.
Filesystem(s) error.
Log(s) script failure –internal error-.
Log archiving failure: Disk or/and TSM.

2. CONCLUSIONS

DB2RS not only adds value to HPSS native metadata integrity monitoring and reporting tools but also ensures that our operational staff are monitoring the health and status of the metadata and thus reducing dramatically the risk of loss of data and respectively the time of recoverability.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives San Francisco, CA September 26-27, 2011 Position Paper

Mark Gary
Lawrence Livermore National Laboratory
mgary@llnl.gov

ABSTRACT / SUMMARY

This position paper addresses the reliability and availability of storage systems. Specifically it introduces LLNL storage best practices in the areas of resilient architectures and daily operations which contribute to enhanced availability, reliability and computational integrity in LLNL's 24x7 HPC centers.

INTRODUCTION

Livermore Computing operates multiple 24x7 "lights on" HPC environments and has done so for over 40 years. Availability, reliability and computational integrity are of paramount concern in the Center due to the tremendous investment in, and the importance of, our HPC machines and the data they generate. This position paper outlines, at a very high level, some of the storage system best practices followed by LLNL. The two focus areas covered in this paper are:

- **Resilient Storage Architectures**: Best practices surrounding the storage hardware architectures employed in the LC and their impact on availability and data integrity.
- **Daily Operations**: Storage system best practices surrounding daily operations (from outages and maintenance to training and communications).

Within these areas I very briefly identify best practices as fodder for Workshop discussion.

Resilient Storage Architectures

Allowing storage operations to continue in the face of failure or outage is critical. Among the hardware architecture best practices followed in the LC are:

- ***Scalable Unit Architecture***

Following the lead of our computing platforms we deploy storage hardware using the concept of a Scalable Unit (SU). An SU is the smallest unit of hardware (storage and associated servers) by which you can grow a storage subsystem. Well identified *identical* SUs allow for ease of repair, maintenance, administration, expansion, and sparing. The purchasing power of buying identical hardware in volume is an added benefit.

- ***Leveraging Compute Platform Hardware***

In our file system and archive environments we, whenever possible, leverage and duplicate the server hardware technologies used on our compute platforms. As in the SU area, this helps ease repair, maintenance, administration, expansion and sparing and takes full advantage of the purchasing power and technology investigation efforts made during platform procurements.

- ***Failover Partners***

The LC has nine very large Lustre file systems. The Object Storage Server (OSS) nodes controlling subsets of disk are architected into failover pairs allowing a failed OSS to have its disk taken over by its healthy partner. This architecture is leveraged constantly and aids not only the case of node failure, but also increases availability during software and firmware deployments and electrical work.

- ***Targeted Use of Uninterruptible Power Supplies (UPS)***

The amount of electrical power required by LC compute and infrastructure hardware makes full coverage with backup power/UPS power economically infeasible. Instead the LC uses its UPS budget in a targeted manner with a concentration on support of metadata services, network infrastructure, and home directory file systems. This strategy focuses on the protection of the most critical data and aides in a rapid return to service of Center operations following a power outage.

- ***Dual Power Sources***

The majority of LC storage infrastructure racks are wired in a redundant manner to be able to survive the planned or unplanned loss of any one electrical subpanel. This is particularly critical in our very dynamic machine room environment during this era of enhanced electrical safety rules.

- ***Archive Dual Copy***

While budgets do not allow us to keep multiple copies of the PetaBytes of simulation data stored in the LC, our HPSS systems do allow a user to direct that dual copies be made of targeted files. The two copies are stored in geographically separated locations cross-Laboratory.

- ***Degraded Mode Archive Operation***

Because of the distributed, multi-level hierarchy architecture of our HPSS archive implementation we are commonly able to provide users with various levels of degraded mode service during outages. In degraded mode not every file is accessible, but typically the most recently written files and those with dual copies can be accessed.

Daily Operations

On a daily basis the LC implements a number of operational best practices surrounding our storage systems including:

- ***"Lights On" Operation***

It is our philosophy, in part driven by the tremendous dollar investment in our computing environment, that the LC provide 24x7x365 customer service and compute availability. Our cross-trained operations staff is always on site monitoring our systems and answering off-hours user questions - around the clock. In the event of an environmental emergency (e.g., loss of cooling) they can immediately react following prescribed/ordered power down procedures. Operations staff are trained in basic file system and archive administration and do hardware repair as well. They have full access to on-call storage system and archive administrators at any hour.

- ***Self-maintenance***

Much of our hardware maintenance is performed by LC employees. This allows us to perform maintenance immediately when a problem occurs, eliminates security escort requirements, and allows us to closely track and learn from system failures. The fact that storage hardware leverages platform hardware procurements means that our personnel need be trained on only a limited number of equipment types.

- ***Hardware/Spare Burn-in***

We have a full hardware spare/RMA center supporting our local maintenance operations. Rather than pulling spare parts off of the shelf, we maintain a burn in environment where we have spare storage hardware under continuous test, exercise, and burn in. When equipment fails, hardware is pulled directly from the burn in environment. Before tape drives are allowed to be placed into service they first undergo a suite of performance tests and integrity tests.

- ***Testbeds***

Livermore Computing has a variety of testbeds in which we test pre-production hardware and software. These testbeds range from single racks, to the well-known multi-vendor Hyperion test environment where software can be tested at scale with thousands of clients.

- ***Data Integrity Checking***

Continuously in the background, the LC runs a tool called DIVT - the Data Integrity Verification Tool. DIVT checks the data integrity of our archive and our parallel file systems by writing known data patterns to files from different platforms, forcing the data to flow through file systems and down to archival tape, and then checking data integrity upon fetch back to the platform. Over the years DIVT has caught data corruption ranging from on-platform component problems to corrupting drive firmware.

- ***Planned Downtimes***

The LC has a philosophy that planned downtimes happen during the work week from Tuesday through Thursday unless particular circumstances dictate otherwise. While this has an impact on interactive users, Center resources are fully subscribed 24x7 including weekends. Our philosophy allows us to have experts from all disciplines on hand in case of problem in order to improve availability. Fridays are avoided to limit the introduction of problems impacting the weekend. Mondays are avoided to allow users to process the results of their weekend runs.

Software rollouts are planned in such a manner as to minimize impact on the programs supported by the Center. We rollout software to our unclassified systems first which allows us to bring outside experts to bear on problems encountered. Recently we leveraged a Six Sigma quality project to improve our software rollout process and reduce the length of planned downtimes.

- ***Impacts and File System Meetings***

Every Monday representatives from every facet of the LC (including Facilities) have a formal meeting to manage any outage or operation that has impact on Center customers or has cross-cutting impact among Center discipline areas. This meeting has tremendous value and allows us to combine outages and plan forward in order to maximize the availability of all center resources including storage. A separate meeting which pulls together Operations staff, storage system administrators, and hardware repair personnel occurs weekly. This meeting improves file system specific communication across all involved disciplines and shifts.

CONCLUSIONS

Large HPC environments are extremely complex. They require that particular attention be paid to operational and architectural storage system best practices in order to ensure availability, reliability and computational data integrity.

* This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-CONF-497278

U.S. Department of Energy Best Practices Workshop on File Systems & Archives
San Francisco, CA
September 26-27, 2011
Position Paper

Evan J. Felix
Pacific Northwest National Laboratory
evan.felix@pnnl.gov

ABSTRACT / SUMMARY

The EMSL Molecular Science Computing team manages 3 main storage systems to provide scientific data storage to the users of EMSL scientific instruments and computing resources. These storage systems are managed in different ways to protect the availability of data and protect from data loss. This position paper will address the reliability and availability of storage systems track of the Best Practices Workshop

The management team has designed and monitors the systems to protect the scientific investment that is stored within the systems. Widely available open-source tools, and home-grown tools are used to monitor and track usage. Communication with users has also been a key element in keeping the systems stable.

INTRODUCTION

The Environmental Molecular Sciences Laboratory(EMSL) at PNNL is a center for scientific research. The building houses many scientific instruments and tools, along with a large computational center. This arrangement of science and computing creates a large amount of scientific data, that we must preserve and store for many years to come. A mix of raw experimental data, processed data, and simulation data is processed and stored with EMSL's file systems and data archive.

EMSL has three main data storage systems, with specific purposes. A home file system for active user codes and data on the cluster, a high speed global file system attached to the large 167 TF Chinook[3] cluster, and a 6+ Pebibyte archive system for long-term storage. These data storage areas are summarized in Table 1.

Table 1

Type	Size	Type	Speed
Home	20 TiB	Lustre	1GB/s
Global temp	270 TiB	Lustre	30GB/s
Long-term archive	4+ PiB	HPSS	200MB/s single stream

All of these file systems are accessible to users directly on cluster login nodes. Home and temp space is available to all cluster nodes.

1. CLUSTER FILE SYSTEMS

The home space and temporary space used for the Chinook cluster are connected directly to the QDR infiniband interconnect. Each system utilizes an active-passive fail-over pair of meta-data servers to manage the file system. The block device for the file system is a RAID1 mirror of two fibre-channel based Virtual RAID5 LUNs. Each system has multiple paths through a switch to each LUN. These storage systems are HP EVA6000 based arrays.

Lustre OST's are also built using a failover pair, but utilize an active-active strategy to balance the load of 8 large LUNs served by the HP EVA technology. Each of these LUNs use a Virtual RAID5 protection scheme. We have been successful in using active-active, as we put all the heartbeat traffic on a very quiet network, which seems to alleviate the dual node power off issue we have seen in many failover solutions. There are 4 servers for the home file system and 38 servers for the temporary file system.

The infiniband connections for the storage servers are balanced across lower-ranked switches of the federated infiniband network. We have also enabled a QOS strategy with OpenSM, using an EMSL created routing algorithm (Down-Up) that has reduced congestion on the network.

Configuration data, including failure states is gathered from all HP EVA systems and stored in the EMSL MASTER database[2]. This database keeps a historical record of all hardware assets, including serial numbers, firmware versions, and status information. A nagios monitoring script can query this database to alert administrators when components fail. Most replacements can be done online, and do not require a file system shutdown. Documentation for all replacement procedures is kept in the system wiki. This database also allows us to look at failure history over the life of the system, and track when components are changed.

The home portion of the file system is backed up on one dedicated node on a daily basis. IBM's Tivoli Storage Manager is used for this purpose. The backup tape system is housed in another building. To perform daily backups a multi-process script was created to keep many streams moving. The temporary space is not backed up.

ARCHIVE SYSTEM

EMSL procured a new HPSS storage system for archive purposes in 2009, and plans to operate it through at least 2017, with planned lifecycle replacements and technology refreshes for the storage components.

The archive system is an HSM based system using the IBM HPSS software stack. We currently have .5 PiB of disk as the first layer of the stack. It consists of one DDN 9900 couplet, serving data to the HPSS mover nodes, data is stored on DirectRAID™ 6 protected LUNS. As data initially moves into the archive it is stored on this disk cache.

The HPSS system has access to an IBM 3584 tape library, which has a mix of LTO4 and LTO5 tapes. Each data block written to tape is stored twice, to protect against tape loss. This duplication policy was implemented as external backups were no longer feasible. The tape library is in another datacenter on the PNNL campus, which is connected with a 2 10Gbits/s network links for a redundant network system.

One key design point we have used for our archive over the last two iterations has been to never lose data. We allow for more downtime and administrator control to accomplish this, and do not require a specific uptime requirement, but do treat it as a production system which should be online as much as possible.

The EMSL MSC team wrote a FUSE[1] based file system for access to the archive, that presents to the users a POSIX-like interface. This system allows us to control more aspects of what a user can do on the system, and increase transfer rates than other tools. It also allows us to 'catch' any file removal actions and move them to a special 'trash' location so that accidental removals are not catastrophic. Since the HPSS unlink commands remove all references to a file in the meta-data store, recovery is very difficult or impossible.

Another management tool we use periodically scans the file system collecting information on users use, and provides to users and administrators information on file counts, and size used by each user. This database also contains historical size of the file system. You can see the size of the file system change in

Figure 1, when we moved from the old archive to the new one as multiple copies were made. The sharp increase in data in the chart is caused by HPSS having amore than one copy of each file. By policy it should have two copies on tape, and may have one in the disk cache as well.

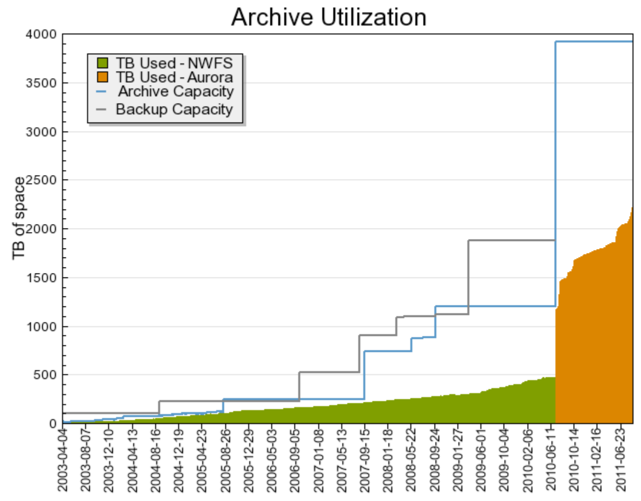


Figure 1

Communication with specific users has been a key aspect of our ability to upgrade and change parts of the archive. Our biggest user by space has assisted in helping us test new changes and been helping in working out any problems before the archive is released back to other users. We also maintain a test system that is built using similar hardware to the production system. This test system allows us to develop and test upgrades before users are on the system.

MANAGEMENT

Much of our success in managing and protecting users data comes from having experienced administrators and programmers directly responsible for the management of storage resources.

When the archive tools provided by the vendor proved in-adequate for our needs, we wrote an appropriate interface to the archive in a few weeks, and added on features, as we needed them. This change to our strategy did not delay our schedule in deploying the archive. And we found when we attended a Users Group meeting for HPSS that others also had similar issues and were very interested in know what we had done, and we were able to share our code with them

When Lustre problems have come up, our extensive knowledge of the internal workings of the Lustre source code has been invaluable in saving, and in one catastrophic case saved us from weeks of backup restores. In our new archive backups are no longer used, and so restoring over a pebibyte of data is no longer an issue.

When a vendor specific monitoring system does not integrate with our monitoring infrastructure, we have been able to write wrappers, or use their low-level API to collect data, and detect failures.

We also maintain a MSC wiki that contains the administrative procedures for handling issues, and routine maintenance tasks. We keep an offline copy of this wiki on a USB drive for reference during complete power or network outages.

One member of our team is also on-call at any time. We have found that having every member of the team being a least front-line response, each member learns basic administration of our critical systems, even when they must bring in other experts to solve unexpected issues.

CONCLUSIONS

The MSC team has found that a deep knowledge of the storage systems that will important scientific data we can design and protect against various failure scenarios and keep the data online and available for our users. Being able to write customized portions of our system allows better integration and management of our resources. This knowledge allows us to be pro-active in finding problems in our system before any data-loss is seen or users experience problems.

REFERENCES

1. FUSE: Filesystem in user space http://fuse.sourceforge.net/
2. Simmons, Chris, Evan Felix, and David Brown. *Understanding a Supercomputer: Utilizing Data Visualization*. Tech.

3. Turner AS, KM Regimbal, MA Showalter, WA De Jong, CS Oehmen, ER Vorpagel, EJ Felix, RJ Rousseau, and TP Straatsma. 2009. *"Chinook: EMSL's Powerful New Supercluster."* *SciDAC Review* 13(Summer 2009):60-69.
4. "DirectRAID - DataDirect Networks." *DDN* | *DataDirect Networks* |. Web. 31 Aug. 2011. http://www.ddn.com/products/directraid

U.S. Department of Energy Best Practices Workshop on File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Venkatram Vishwanath, Mark Hereld and Michael E. Papka
Argonne National Laboratory
<venkatv, hereld, papka>@mcs.anl.gov

ABSTRACT

The performance mismatch between the computing and I/O components of current-generation HPC systems has made I/O a critical bottleneck for scientific applications. It is therefore crucial that software take every advantage available in moving data between compute, analysis, and storage resources as efficiently as networks will allow. Currently available I/O system software mechanisms often fail to perform as well as the hardware infrastructure would allow, suggesting that improved optimization and perhaps adaptive mechanisms deserve increased study.

We describe our experiences with GLEAN – a simulation-time data analysis and I/O acceleration infrastructure for leadership class systems. GLEAN improves the I/O performance, including checkpointing data, by exploiting network topology for data movement, leveraging data semantics of applications, exploiting fine-grained parallelism, incorporating asynchronous data staging, and reducing the synchronization requirements for collective I/O.

INTRODUCTION

While the computational power of supercomputers keeps increasing with every generation, the I/O systems have not kept pace, resulting in a significant performance bottleneck. The *ExaScale Software Study: Software Challenges in Extreme Scale Systems* explains it this way: ``Not all existing applications will scale to terascale, petascale, or on to exascale given current application/architecture characteristics" citing ``I/O bandwidth" as one of the issues. On top of this, one often finds that existing I/O system software solutions only achieve a fraction of quoted capabilities.

We have developed an infrastructure called GLEAN [1,2] to accelerate the I/O of applications on leadership systems. We are motivated to help increase the scientific output of leadership facilities. GLEAN provides a mechanism for improved data movement and staging for accelerating I/O, interfacing to running simulations for co-analysis, and/or an interface for in situ analysis via a zero to minimal modification to the existing application code base. GLEAN has scaled to the entire infrastructure of the Argonne Leadership Class Facility (ALCF) comprising of 160K Intrepid IBM Blue Gene/P (BG/P) cores and demonstrated multi-fold improved with DOE INCITE and ESP applications. We discuss some of the lessons learned which could be considered for best practices on file systems and archives.

OUR POSITION

Based on our experiences with GLEAN, we believe the useful components to improve the I/O performance on leadership class systems include topology-aware data movement, leveraging data semantics, incorporating asynchronous data staging, leveraging fine-grained parallelism, and non-intrusive integration with applications. We briefly elucidate these.

Topology-aware Data Movement: As we move towards systems with heterogeneous and complex network topologies, effective ways to fully exploit their heterogeneity is critical. The IBM BG/P has five different networks with varying throughputs and topologies. The 3D torus interconnects a compute node with its six neighbors at 425 MB/s over each link. In contrast, the tree network is a shared network with a maximum throughput of 850 MB/s to the I/O nodes. The tree network is the only way to get to the I/O nodes in order to perform I/O. BG/Q is expected to have a more complex network topology. Similarly, several other Top-500 supercomputers have complex topologies. As seen in Figure 1, by leveraging the various network topologies, in GLEAN, we achieve up to 300-fold improvement in moving data out from the BG/P system. Another critical aspect is that our data movement mechanism uses reduced synchronization mechanisms wherein only neighboring processes need to co-ordinate their I/O. This is critical as we move towards future systems with millions of cores.

Fine-grained Parallelism: GLEAN's design employs a thread-pool wherein each thread handles multiple connections via a poll-based event multiplexing mechanism. This is critical in future many-core systems with low clock-frequency per core, where multiple threads are needed to drive the 40 Gbps and higher network throughputs per node to saturation.

Asynchronous data staging refers to moving the application's I/O data to dedicated nodes and next writing this out to the filesystem asynchronously while the application proceeds ahead with its computation. Asynchronous data staging helps satisfy the bursty nature of application I/O common in computational science and blocks the simulation's computation only for the duration of copying data from the compute nodes to the staging nodes. Data staging also significantly reduces the number of clients seen by the parallel filesystem, and thus mitigates the contention including locking overheads for the filesystem. Staging mitigates the variability in I/O performance seen in shared filesystems on leadership systems when accessed concurrently by multiple applications.

Leveraging Application Data Models: I/O system software typically use stream of bytes and files to deal with an application's data. A key design goal in GLEAN is to make application data models a first-class citizen. This enables us to apply various analytics to the simulation data at runtime to reduce the data volume written to storage, transform data on-the-fly to meet the needs of analysis, and enable various I/O optimizations leveraging the application's data models. Toward this effort, we have worked closely with FLASH, an astrophysics application, to capture its adaptive mesh refinement (AMR) data model. We have interfaced with PHASTA, which uses an adaptive unstructured mesh, to make unstructured grids supported in GLEAN, and with S3D, a turbulence simulation, to capture it's structured grid model. We have worked with many of the most common HPC simulation data models ranging from AMR grids to unstructured adaptive meshes.

Non-Intrusive Integration with Applications: Application scientists are very interested in I/O solutions wherein they can get the added performance improvements without having to change their simulation code (or with minimal changes). To achieve this, we have mapped Parallel-netCDF and hdf5 APIs, commonly used high-level I/O libraries in simulations, to relevant GLEAN APIs, thus enabling us to non-intrusively interface with simulations using pnetcdf and hdf5.

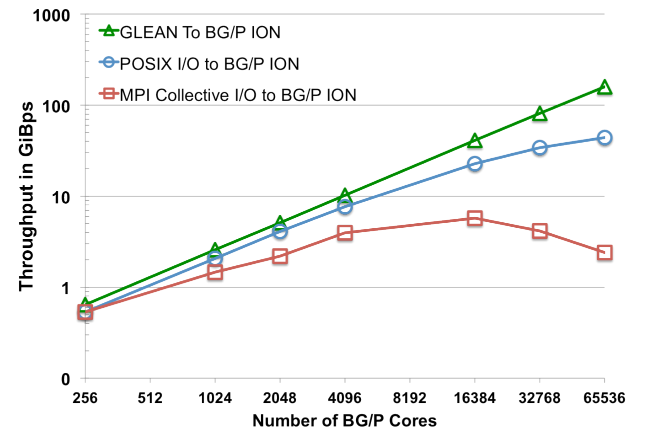


Figure 1: Strong scaling performance of the I/O mechanisms to write 1 GiB data to the BG/P IONs (log-log scale) on ALCF infrastructure

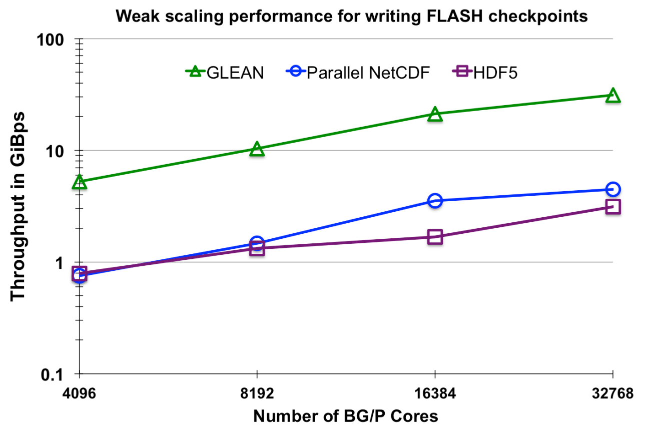


Figure 2: Weak scaling results for writing FLASH checkpoint data

REFERENCES

1. V. Vishwanath, M. Hereld, V. Morozov, and M. E. Papka, "*Topology-aware data movement and staging for I/O acceleration on Blue Gene/P supercomputing systems*", To appear in IEEE/ACM International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2011), Seattle, USA, November 2011.

2. V. Vishwanath, M. Hereld, and M. E. Papka, "*Simulation-time data analysis and I/O acceleration on leadership-class systems using GLEAN*", To appear in IEEE Symposium on Large Data Analysis and Visualization (LDAV), Providence, RI, USA, October 2011.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

HPC Data Storage Strategy at the UK Atomic Weapons Establishment (AWE)

Mark Roberts
AWE High Performance Computing
mark.roberts@awe.co.uk

Paul Tomlinson
AWE High Performance Computing
paul.tomlinson@awe.co.uk

ABSTRACT / SUMMARY

AWE has adopted a resilient and globally accessible storage model to support concurrent desktop and compute cluster access. We now provide a global persistent multi-tiered storage which has enhanced usability and reliability. Increasing pressure on budgets has recently focused efforts to reduce and consolidate the directly-attached parallel cluster file system into several global scratch file systems cross mounted on all compute clusters.

INTRODUCTION

Historically, AWE has focused on one or more compute clusters, each with its own local parallel scratch file system and global persistent storage provided by commodity Network Attached Storage (NAS) filers or in-house Network File System (NFS) servers.

Following a major facility issue a few years earlier, which resulted in the sole persistent data store, based on IBMs General Parallel File System (GPFS), being corrupted and had to be restored (very slowly) from backup, it was decided to upgrade to a resilient GPFS cluster. This was designed to provide multi-site resilience, protecting from loss of either site.

This has allowed us to maintain native multicluster GPFS access to the numerous cluster log-in nodes, visualisation clusters, and secure NFSv4 access direct to the desktop.

COMMON ENVIRONMENT

A common name space, providing a consistent view of file systems on desktops, compute and visualization clusters, helps users locate their data and has encouraged well structured work flow with respect to makefiles, shared libraries and code areas.

This, combined with other common environment features, has aided the trend towards more local prototyping, with easier scaling up onto larger platforms.

DESKTOP ENHANCEMENT

The Hierarchical Storage Managed (HSM) file systems are now exposed to desktops, running file managers and search utilities, that scan, index and try to determine type by content, all potentially triggering unwanted retrievals, which can block interactive access until the recall from tape completes.

To mitigate this, we modified the KDE3 file manager and GNU utilities (*ls, find, stat, file)* to be aware that a file is migrated based on the naive concept that a migrated file has a non-zero file size with zero data blocks. Users are given visual cues with syntax highlighting or icon overlays, along with extra options for handling such files. Preview operations that would generate multiple recalls are skipped. Obtaining the migration state via an API or extended attributes that could propagate through software and network protocols would be preferable. However the simple approach has worked well in our environment.

FILESYSTEM USAGE TRENDS

For many years, we recorded per-user GPFS file system usage with a simple POSIX *find* command in conjunction with the HSM *dsmls* command. This was highly inefficient, taking over 24 hours to complete a scan, as well as adding unnecessary CPU and I/O load on the Tivoli Storage Manager (TSM) server.

We now utilize the GPFS Information Lifecycle Management (ILM) interface to scan the file systems and output records containing the extended information such as name, size, access, modify and create time. The resulting data file is processed and imported into a MySQL database. This allows for fine granular analysis at the user and file system level day by day or over a time period.

Currently, with 35 million objects, the new method takes under 20 minutes. We believe that providing such granular information influences users' data storage behavior for the better, and lets us quickly identify unreasonable or unintended usage when problems occur. It also provides long-term trend information to aid future file system capacity planning and procurement.

We have also engaged with Lustre developers to see if future Lustre releases can incorporate a similar capability.

DECOUPLED PARALLEL FILESYSTEMS

Traditionally, as part of a compute cluster procurement, we purchased storage for a dedicated parallel file system to provide the localised fast bandwidth required by codes.

With the potential of cluster lifetimes becoming shorter due to the the ever increasing pace of technology releases, the cost of the disk infrastructure is now becoming a factor. Once our clusters were decommissioned the disk was also removed and disposed of. With a recent capability cluster procurement the local parallel file system hardware accounted for approximately 15% of the total expenditure. The demand for storage is increasing, so as the total of memory and cores on clusters the associated storage costs may have a larger impact.

We have decided to "decouple" the local parallel file system which allows the storage to be used by multiple clusters. This gives the freedom to either use the cost reduction by increasing the compute size or directing the saving elsewhere. One immediate advantage of no directly attached storage is more rapid initial cluster deployment and, possibly, regular scheduled maintenance without affecting access to the data via other clusters.

The community gain improved useability by not having to transfer the data between the local parallel file systems on the clusters and then to persistent storage.

By extending the concept of resilience to global scratch with a global scratch file system cluster in each facility (three planned) we can now factor in scheduled downtime and upgrades to a chosen global scratch file system cluster more effectively.

DATA AND FS AWARE SCHEDULING

It is our intention that each compute cluster should use by default the global scratch file system in its local facility.

Users may, however, wish to use another global scratch file system in a different building or use the "local" global scratch in conjunction with it. In order to prevent jobs failing due to an unavailable global scratch, we are investigating the concept of storage as a consumable resource in the same manner as a node or cores is used today.

By integrating awareness of global storage into the scheduler/resource manager, jobs may be prevented from failing prematurely. Also when global scratch or persistent storage clusters are scheduled for maintenance then scheduler system reservations can be placed on the file system preventing jobs from being dispatched, if the job requested that file system as a resource.

DATA EXPLOSION

With an increasing amount of scratch, persistent storage and upgraded network links, it becomes relatively easy for the user to copy everything everywhere. This leads to wasted bandwidth, disk storage and tape backups due to duplicated data. File system de-duplication on persistent storage may be possible but ultimately undesirable.

With early MPP clusters it was often quicker to move data from disk or recall from off-line storage than regenerate the data. With the large fast clusters available today it may, in some cases, be quicker and cheaper to save network, disk, and tape resources by regenerating data. This is ultimately a decision that only the user is best placed to make but having to weight up the impact on QA reproducibility and provenance.

Existing compute cluster parallel file systems were designed with the ability to hold four times the amount of system memory from a OS initiated checkpoint. The majority of the mainstream codes are now using restart dumps generated via an in-house I/O library. Code users can then choose whether to perform a full state restart or focus on only saving selected data within the run for analysis. Intelligent software-based restarts greatly reduce the amount and bandwidth required and could allow considerable cost savings, but non-restartable third party codes remain a barrier.

CONCLUSION

Implementation of a fast, secure, exported and resilient global parallel file system for persistent and archive storage has proved invaluable for unifying compute resources at all scales. However the ease of accessibility has created some additional problems and raised user expectations, requiring adaptation of the user interface. We are now exploring a similar approach for efficient global scratch storage.

U.S. Department of Energy Best Practices Workshop on File Systems & Archives:[*] Usability at Los Alamos National Lab[†]

John Bent
Los Alamos National Lab
johnbent@lanl.gov

Gary Grider
Los Alamos National Lab
ggrider@lanl.gov

Abstract

There yet exist no truly parallel file systems. Those that make the claim fall short when it comes to providing adequate concurrent write performance at large scale. This limitation causes large usability headaches in HPC computing.

Users need two major capabilities missing from current parallel file systems. One, they need low latency interactivity. Two, they need high bandwidth for large parallel IO; this capability must be resistant to IO patterns and should not require tuning. There are no existing parallel file systems which provide these features. Frighteningly, exascale renders these features even less attainable from currently available parallel file systems. Fortunately, there is a path forward.

1 Introduction

High-performance computing (HPC) requires a tremendous amount of storage bandwidth. As computational scientists push for ever more computational capability, system designers accommadate them with increasingly powerful supercomputers. The challenge of the last few decades has been that the performance of individual components such as processors and hard drives as remained relatively flat. Thus, building more power supercomputers requires that they be built with increasing numbers of components. Problematically, the mean time to failure (*MTTF* of individual components has over remained relatively flat over time. Thus, the larger the system, the more frequent the failures.

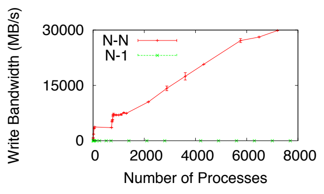
Traditionally, failures have been dealt with by periodically saving computational state onto persistent storage and then recovering from this state following any failure (*checkpoint-restart*). The utilization of systems is then measured using *goodput* which is the percentage of computer time that is spent actually making progress towards the completion of the job. The goal of system designers is therefore to maximize goodput in the face of random failures using an optimal frequency of checkpointing.

Determining checkpointing frequency should be straight-forward: measure MTTF, measure amount of data to be checkpointed, measure available storage bandwidth, compute checkpoint time, and plug it into a simple formula [3]. However, measuring available storage bandwidth is not as straightforward as one would hope. Ideally, parallel file systems could achieve some consistent percentage of the hardware capabilities; for example, a reasonable goal for a parallel file system using disk drives for storage would be to achieve 70% of the aggregate disk bandwidth. If this were the case, then a system designer could simply purchase the necessary amount of storage hardware to gain sufficient performance to minimize checkpoint time and maximize system goodput. However, there exist no currently available parallel file systems that can provide any such performance level consistently.

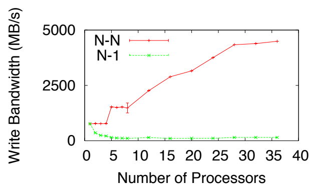
2 Challenges

Unfortunately, although there are some that can, there are many IO patterns that *cannot* achieve any consistent percentage of the storage capability. Instead, these IO patterns achieve a consistently low performance such that their percentage of hardware capability diminishes as more hardware is added! For example, refer to Figures 1a, 1b, and 1c, which show that writing to a shared file, *N-1*, achieves consistently poor performance across the three major parallel file systems whereas the bandwidth of writing to unique files, *N-N*, scales as desired with the size of the job. The flat lines for the N-1 workloads actually show that there is no amount of storage hardware that can be purchased: regardless of size, the bandwidths remain flat. This is because the hardware is not at fault; the performance flaw is within the parallel file systems which cannot incur massively concurrent writes and maintain performance. The challenge is due

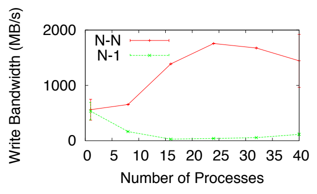
[*]San Francisco, CA; September 26-27, 2011
[†]LANL Release LA-UR 11-11416



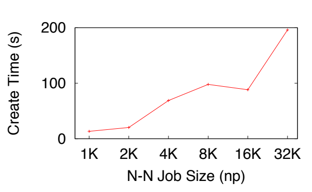
(a) PanFS



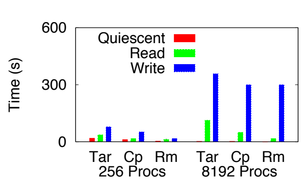
(b) GPFS



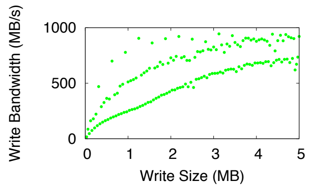
(c) Lustre



(d) Metadata



(e) QoS



(f) Tuning

Figure 1: **Usability Challenges.** *These graphs address the key usability challenges facing today's users of HPC storage systems. The top three graphs demonstrate the large discrepancy between achievable bandwidth and scalability using N-N and N-1 checkpoint patterns on three of the major HPC parallel file systems. The bottom left graph shows the challenge of metadata creation at large job size, the bottom middle shows how the notion of interactivity is a cruel joke when small file operations are contending with large jobs performing parallel IO, and finally, the bottom right graph shows the reliance on magic numbers that plagues current parallel file systems.*

to maintaining data consistency which typically requires a serialization of writes.

An obvious solution to this problem is for all users to always perform N-N file IO in which every process writes to a unique file. This approach does not come without trade-offs however. One is a performance limitation at scale and the other is a reduction in usability as will be discussed later in Section 3.

The system problem is the massive workload caused by by large numbers of concurrent file creates when each process opens a new file. Essentially this causes the same exact problem on parallel file systems as does writing in an N-1 pattern: concurrent writes perform poorly. In this case, the concurrent writing is done to a shared directory object. These directory writes are handled by a metadata server; no current production HPC parallel file system supports distributed metadata servers. As such, large numbers of directory writes are essentially serialized at a single metadata server thus causing very large slow-downs during the create phase of an N-N workload as is shown in Figure 1d.

3 Implications for Usability

This causes large usability headaches for LANL users. All of the large computing projects at LANL are well-aware of, and dismayed by, these limitations. All have incurred large opportunity costs to perform their primary jobs by designing around these limitations or paying large performance penalties. Many create archiving and analysis challenges for themselves by avoiding writes to shared objects by having each process in large jobs create unique files. Some have become parallel file system experts and preface parallel IO by doing complicated queries of the parallel file system in order to rearrange their own IO patterns to better match the internal data organization of the parallel file system.

3.1 Tuning

Many users have learned that parallel file systems have various *magic numbers* which correspond to IO sizes that achieve higher performance than other IO sizes. Typically these magic numbers correspond with various object sizes in the parallel file system ranging from a disk block to a full RAID stripe. The difference between poorly performing IO sizes and highly performing IO sizes is shown in Figure 1f which was produced using LBNL's PatternIO benchmark [8]. Also, this graphs seems to merely show that performance increases with IO size, a closer examination shows that there are many small writes that perform better than large writes. In fact, a close examination reveals three distinct curves in this graph: the bottom is IO sizes matching no magic numbers, the middle is for IO sizes in multiples of the object size per storage device, and the upper is for IO sizes in multiples of the full RAID stripe size across multiple storage devices.

The implication of this graphs is that those users who can discover magic numbers and then use those magic numbers can achieve much higher bandwidth than those users who cannot. Unfortunately, both discover-

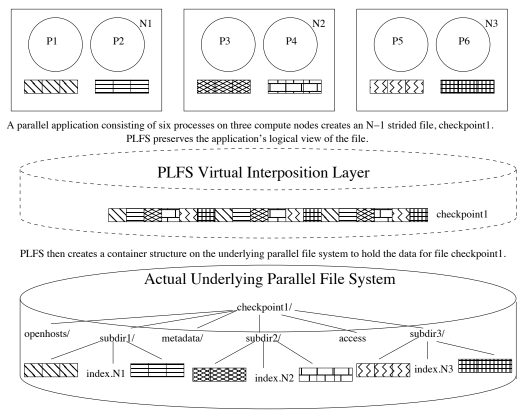

N1
P1
P2
N2
P3
P4
N3
P5
P6
A parallel application consisting of six processes on three compute nodes creates an N−1 strided file, checkpoint1.
PLFS preserves the application's logical view of the file.
PLFS Virtual Interposition Layer
checkpoint1
PLFS then creates a container structure on the underlying parallel file system to hold the data for file checkpoint1.
Actual Underlying Parallel File System
checkpoint1/
openhosts/
subdir1/
metadata/
subdir2/
access
subdir3/
index.N1
index.N2
index.N3

Figure 2: **PLFS Architecture.** *This figure shows how PLFS maintains the user's logical view of a file which physically distributing it into many smaller files on an underlying parallel file system.*

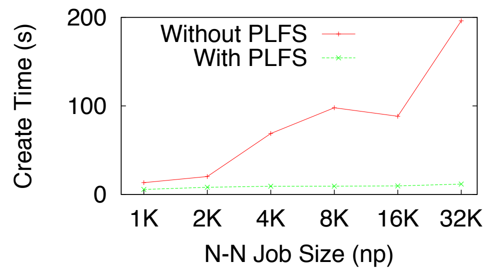


Figure 3: **Addressing Metadata Challenge.** *This graph shows how distributed metadata keeps create rates manageable at large scale.*

ing and exploiting magic numbers is difficult and often intractable. Magic numbers differ not only on different parallel file systems (*e.g.* from PanFS to Lustre) but also on different installations of the same file system. Tragically, there is no simple, single mechanism by which to extract magic numbers from a file system.

We have a user at LANL who executes initialization code which first queries *statfs* to determine the file system *f_type* and then, based on which file system is identified, then executes different code for each of the three main parallel file systems to attempt to discover the magic number for that particular installation. Once discovering this value, the user then reconfigures their own, very complicated, IO library to issue IO requests using the newly discovered magic number. Of course, most users would not prefer to jump through such hoops, and frankly, many users should not be trusted with low-level file system information. Not because they lack intelligence but because they lack education; they are computational scientists who should not be expected to become file system experts in order to extract reasonable performance from a parallel file system.

Of course, even if all users could easily discover magic numbers, they could not all easily apply them. For example, many applications do adaptive mesh refinement in which the pieces of the distributed data structures are not uniformly sized: neither in space nor in time. This means that users looking for magic numbers will need some sort of complicated buffering or aggregation. An additional challenge is that magic numbers are not as easy as merely making the individual IO operations be of a particular size; they must also be correctly aligned with the underlying object sizes. So not only must users attempt to size operations correctly, they must also attempt to align them correctly as well. There are other approaches to address this problem such as collective buffering [10] in MPI-IO. As we will show later in Section 4.1, collective buffering is beneficial but is not a complete solution.

3.2 Quality of Service

Finally, although checkpoint-restart is a dominant activity on the storage systems, obviously it is not the only activity. Computational science produces data which must then be explored and analyzed. As the output data is stored on the same storage system which services checkpoint-restart, data exploration and analysis workloads can content with checkpoint-restart workloads. As is seen in Figure 1e, the checkpoint-restart workloads can wreak havoc on interactive operations. In this experiment, the latency of small file operations, such as untarring a set of files, copying that same set of files, and then removing the files, was measured during periods of quiescence and then compared to the latency of those same operations when they were contending with large parallel jobs doing a checkpoint write and a restart read. The most painful latency penalties are seen when the operations contend with a 8192 process job doing a checkpoint write.

4 Path Forward

There are many emerging technologies, ideas, and potential designs that offer hope that these challenges will be addressed in time for the looming exascale era.

4.1 PLFS

Our earlier work in SC09 [2] plays a prominent role in our envisioned exascale IO stack. That work showed how PLFS makes all N-1 workloads achieve the performance of N-N workloads and also how PLFS removes the need for tuning applications to the underlying system (*i.e.* in PLFS, every number is a magic number!). Those results will not be repeated here but suffice it to say that they

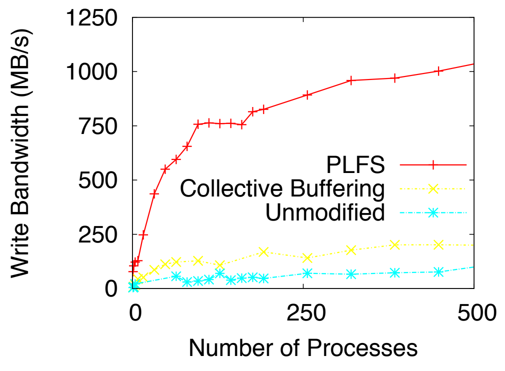


Figure 4: **Collective Buffering.** *This graph shows that collective buffering may not be sufficient for many workloads.*

eliminate the challenges show in Figures 1a, 1b, 1c, and 1f. From a usability perspective, PLFS is an important contribution: in addition to removing the need for IO tuning, PLFS is transparently accessible by *unmodified* applications using either POSIX IO or the MPI IO libraries.

Note that collective buffering [10] is another approach to dealing with the thorny problem of magic numbers. Figure 4 shows that, for one particular workload, collective buffering is an improvement over an unmodified approach to IO but underperforms the bandwidth obtainable using PLFS. In fairness, however, we are not collective buffering experts and perhaps collective buffering could be tuned to achieve much higher bandwidth. Ultimately though, our usability goal is to remove file system and parallel IO tuning from the user's purview.

Figure 2 shows the architecture of PLFS and how it preserves the user's logical view of a file while physically striping the data into many smaller files on an underlying parallel file system. This effectively turns all large parallel workloads into N-N workloads. Of course, as we saw in Figure 1d, even N-N workloads suffer at very large scale. Additionally, we know that this performance degradation is due to an overloaded metadata server which will destroy interactive latency as we saw in Figure 1e.

Borrowing ideas from GIGA+ [7], PLFS now addresses these challenges as well. Recent versions of PLFS (since 2.0) can stripe the multiple physical files comprising a logical file over multiple file systems. In the case where each file system is served by a different metadata server, this distributes metadata load very effectively as can be seen in Figure 3 which is the same as Figure 1d but with an added line showing how metadata distribution within PLFS can remove metadata bottlenecks. Note that the workload shown was run using an N-N workload. Although PLFS was originally designed for N-1 workloads, this new functionality will allow PLFS to address metadata concerns for all exascale checkpoint workloads.

5 Redesigning the IO Stack

PLFS has proven to be a very effective solution for current IO challenges: it allows all workloads to easily achieve a consistently high percentage of the aggregate hardware capability.

PLFS is not sufficient however to solve the looming exascale IO challenges before us. Recent work [9] shows that the checkpointing challenge is becoming increasingly difficult over time. The checkpoint size in the exascale era is expected to be around 32 PB. To checkpoint this in thirty minutes (a decent rule of thumb) requires 16 TBs of storage bandwidth. Economic modeling shows that current storage designs would require an infeasible 50% of the exascale budget to achieve this performance.

5.1 Burst buffer

We must redesign our hardware stack and then develop new software to use it. Spinning media (*i.e.* disk) by itself is not economically viable in the exascale era as it is priced for capacity but we will need to purchase bandwidth. Additionally, the storage interconnect network would be a large expense. Thus far, we have required an external storage system for two main reasons: one, sharing storage across multiple supercomputers improves usability and helps with economies of scale; two, embedding spinning media on compute nodes decreases their MTTF.

Our proposal is to make use of emerging technologies such as solid state devices, *SSD*. This media is priced for bandwidth and for low latency so the economic modeling shows it is viable for our bandwidth requirements. Additionally, the lack of moving parts is amenable to our failure models and allows us to place these devices within the compute system (*i.e.* not on the other side of the storage network). Unfortunately, being priced for bandwidth means these devices cannot provide the storage *capacity* that we require. We still require our existing disk-based parallel file systems for short-term capacity needs (long-term capacity is served by archival tape systems not otherwise discussed here).

We propose adding these devices as a new layer in our existing storage hierarchy between the main memory of our compute nodes and the spinning disks of our parallel file systems; we call this interposition of SSD a *burst buffer* as they will absorb very fast bursts of data and serve as an intermediate buffer to existing HPC parallel file systems. This is not a new idea and is commonly suggested as a solution to the well-known latency gap between memory and disk. Our proposal however is how to specifically incorporate these burst buffers into the existing HPC storage software stack.

5.2 E Pluribus Unum

Our envisioned software stack incorporates many existing technologies. The SCR [6] software is a perfect candidate for helping schedule the burst buffer traffic and to enable restart from neighboring burst bufers within the compute nodes. However, we envision merging SCR and PLFS to allow users to benefit from PLFS's capability to handle both N-1 and N-N workloads and to allow use by unmodified application.

We have already add PLFS as a storage layer within the MPI IO library. This library has many important IO optimizations in addition to collective buffering described earlier. One such optimization is available using *MPI_File_set_view*. This is an extremely nice feature from a usability perspective. This is clear when we consider what computational scientists are doing: they stripe a multi-dimensional data structure representing some physical space across a set of distributed processors. Dealing with these distributed multi-dimensional data structures is complicated enough without even considering how to serialize them into and out of persistent storage. *MPI_File_set_view* lesses this serialization burden; by merely describing their distribution, the user then transfers the specific serialization work to the MPI IO library.

Note that other data formatting libraries such as HDF [1], Parallel netCDF [4], SCR , and ADIOS [5] provide similar functionality and have proven very popular as they remove computer science burdens from computational scientists. These data formatting libraries are the clear path forward to improve usability of HPC storage. However, they will not work in their current form on burst buffer architectures. We envision adding our integrated PLFS-SCR storage management system as a storage layer within these data formatting libraries just as we have done within the MPI IO library. A key advantage of a tight integration between PLFS-SCR and these data formatting libraries is that semantic information about the data can be passed to the storage system thus enabling semantic retrieval.

5.3 In situ data analysis

There are two key features of our proposal that enable *in situ* data analysis. The first is that the burst buffer architecture embeds storage much more closely to the compute nodes which drastically reduces access latencies for both sequential and random accesses. The second is that because the data has been stored using data formatting libraries, semantic retrieval of data is possible. This means that we can more easily attempt to co-locate processes within the analysis jobs close to the burst buffers containing the desired data. Finally, even when the data is not available on a local burst buffer, we can take advantage of the low-latency interconnect network between the compute nodes to transfer data between burst buffers as needed.

6 Conclusion

In this proposal, we have described how current usability of HPC storage systems is hampered by two main challenges: poor performance for many large jobs, and occasional intolerably slow interactive latency. We have offered PLFS as a solution for these challenges on today's systems.

Finally, we point out the inability of PLFS to address exascale challenges by itself. We then offer a proposal for integrating PLFS with a burst buffer hardware architecture PLFS and a set of other existing software packages as one path towards a usable and feasible exascale storage solution.

References

[1] The HDF Group. http://www.hdfgroup.org/.

[2] J. Bent, G. Gibson, G. Grider, B. McClelland, P. Nowoczynski, J. Nunez, M. Polte, and M. Wingate. Plfs: a checkpoint filesystem for parallel applications. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC '09, pages 21:1–21:12, New York, NY, USA, 2009. ACM.

[3] J. T. Daly. A higher order estimate of the optimum checkpoint interval for restart dumps. *Future Gener. Comput. Syst.*, 22(3):303–312, 2006.

[4] J. Li, W. keng Liao, A. Choudhary, R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel, B. Gallagher, and M. Zingale. Parallel netcdf: A high-performance scientific i/o interface. *SC Conference*, 0:39, 2003.

[5] J. F. Lofstead, S. Klasky, K. Schwan, N. Podhorszki, and C. Jin. Flexible io and integration for scientific codes through the adaptable io system (adios). In *CLADE '08: Proceedings of the 6th international workshop on Challenges of large applications in distributed environments*, pages 15–24, New York, NY, USA, 2008. ACM.

[6] A. Moody, G. Bronevetsky, K. Mohror, and B. R. d. Supinski. Design, modeling, and evaluation of a scalable multi-level checkpointing system. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–11, Washington, DC, USA, 2010. IEEE Computer Society.

[7] S. V. Patil, G. A. Gibson, S. Lang, and M. Polte. GIGA+: Scalable Directories for Shared File Systems. In *Petascale Data Storage Workshop at SC07*, Reno, Nevada, Nov. 2007.

[8] Rajeev Thakur. Parallel I/O Benchmarks. http://www.mcs.anl.gov/ thakur/pio-benchmarks.html.

[9] B. Schroeder and G. Gibson. A large scale study of failures in high-performance-computing systems. *IEEE Transactions on Dependable and Secure Computing*, 99(1), 5555.

[10] R. Thakur and E. Lusk. Data sieving and collective i/o in romio. In *In Proceedings of the Seventh Symposium on the Frontiers of Massively Parallel Computation*, pages 182–189. IEEE Computer Society Press, 1999.

PSI – A High Performance File Transfer User Interface

Mark A. Roschke
High Performance Computing Division,
Los Alamos National Laboratory
mar@lanl.gov

C. David Sherrill
High Performance Computing Division,
Los Alamos National Laboratory
dsherril@lanl.gov

Abstract

Transferring and maintaining large datasets requires parallel processing of both data and metadata for timely execution. This paper describes the work in progress to use various processing techniques, including multi-threading of data and metadata operations, distributed processing, aggregation, and conditional processing to achieve increased transfer performance for large datasets, as well as increased rates for metadata queries and updates.

1. Introduction

Ever-increasing computing capabilities result in ever-increasing data sets to be transferred. Such data sets can consist primarily of large files, many small files, or both. Transferring data sets with large files requires an emphasis on parallel file transfer, utilizing as much bandwidth as possible. And it is in this area that the majority of data parallel transfer development has occurred. But, it is no longer rare for a user to generate data sets of 100,000 to one million files. And when data sets reach this size, it is imperative that support be provided for high performance metadata operations, not only in support of file transfer, but also to support browsing and maintaining the data set.

2. Overview of PSI

The Parallel Storage Interface (PSI) is a data transfer user interface designed to provide high speed transfer for large data sets, with a special emphasis on utilizing as many resources as possible for a single user request. Developed by the authors, PSI is the main user interface to the High Performance Storage System (HPSS) at Los Alamos National Laboratory. This paper describes the efforts to provide a full-featured data transfers capability for archival transfer, local transfer, and wide area host-to-host transfer, providing both high-speed data transfer as well as high-speed metadata processing.

PSI uses a parallel workflow model for processing both data and metadata. Work is parallelized and scheduled on available server and client resources automatically, using a priority and resource-based approach. Optimization is performed automatically, including areas such as parallelization, optimized tape transfer, load leveling, etc.

3. Unix syntax and Semantics

PSI utilizes UNIX-like syntax and semantics. For example, the following commands are available for data transfer and manipulation of file attributes: **cd**, **chmod**, **chgrp**, **cp**, **du**, **find**, **grep**, **ls**, **mkdir**, **mv**, **rm**, **rmdir**, and **scp**.

4. Multi-mode Operation

PSI offers three modes of operation, providing the same syntax, semantics, and look and feel for the three most frequently used data transfer situations, which are 1) local transfer, 2) archival transfer, and 3) host-to-host transfer. The particular interface command determines the context of the specified commands. For example,

sh	**cp –R a b**	copy files locally
psiloc	**cp –R a b**	parallel copy locally
psi2ccc	**cp –R a b**	parallel copy on cluster **ccc**
psi	**cp –R a b**	parallel copy in the archive

This approach provides a consistent look and feel, allowing the user to move between the 3 major transfer situations, eliminating the time necessary to learn the command set for each situation.

5. Automatic Optimization

The general design approach for PSI is that the user simply specifies the files to be operated on, PSI determines the resources available to the command, and then executes the command, with all optimization being performed automatically, including such features as adjusting all types of thread counts dynamically, optimizing the order of any data transfers to/from tape, assignment of operations across multiple hosts (including load leveling), and splitting large transfers across hosts when appropriate.

To support automatic optimization, all activity within PSI is controlled using a priority-based resource management scheme, limiting the amount of bandwidth and memory that each type of activity can consume. Scheduling of activities such as file transfers are performed via an internal job scheduler, which dispatches activities across available hosts in an optimal order, load leveling all activities as necessary.

6. Conditional Transfers

To address occurrences of interrupted transfers as well as that of newly arriving (or updated) data within a data set, PSI can scan both the input tree and output tree, examining file attributes to determine which files need to be transferred. This feature alone can routinely save hours of time that would be spent on re-transferring the entire tree.

7. Parallel Archival Tarring Option

When transferring to the archive, the user can select the PSI tar option, which automatically utilizes parallel tar transfers to/from the archive. The parallel tar capability in PSI typically constructs one or more tar files per directory, preserving the original tree structure. Large files are normally transferred un-tarred to the archive. Multiple tar processes are load leveled across available hosts, providing scalable multi-host performance, even for small files.

The archive namespace is extended into the tar files present, utilizing the index file that is stored with each tar file. This namespace extension prevents the archive from becoming a large black box of data. The user can browse through the original tree, and execute such commands as **ls**, **find**, **grep**, **rm**, and **scp** with references to files within the tar files, and can also utilize globbing (i.e. wild cards) in such references. Conditional transfers are also supported, so that newly arrived files can be placed within new tar files in a directory. In addition, commands such as **scp**, **chmod**, **grep**, **ls**, and **rm** are specially aware of the tar files, and can take advantage of operating on whole tar files when feasible.

8. Techniques to Increase Performance

The general approach chosen involves the use of parallel data and metadata processing, automatic optimized file aggregation and de-aggregation, and conditional operations when feasible. Combining these three features provides a variety of performance increases. For example, multi-threading to a degree of 40 threads might increase performance by a factor of 30, while operating on a file aggregate of 1000 files can provide a performance boost of up to 300. Conditional operations can provide a factor of 20 or so. By combining these three features, performance gains of over 1,000 have been observed, as outlined below.

9. Multiphase Parallel Work Flow

To facilitate efficient control of the various steps required to execute user requests in a parallel fashion, For example, tasks are organized into phases, e.g. 1) stat source files, 2) stat destination files, 3) transfer files. Work progresses though each phase. Each phase can consist of many threads, each requiring different resources. Achieving high performance in processing metadata requires a reasonably high degree of parallelism; typical thread counts for all three phases is 100 to 200, depending upon the mix of metadata and file transfer operations being performed.

10. Areas of Performance Increase

Work at increasing performance has fallen in two general areas – increasing parallelism, and decreasing latency with the latter area receiving the most effort. Increasing parallelism generally falls in the predictable categories of more threads, and more hosts,
with some miscellaneous optimization applied to areas such as when to transfer large files across nodes, etc.

Effort to decrease latency has been largely in the area of various types of aggregation, namely 1) data aggregation, 2) control aggregates, 3) metadata query aggregates, and 4) metadata update aggregates.

Metadata query work has involved experiments with striping directory queries across multiple hosts, with an eye toward support of massive directories (directories with greater than 50,000 files).

Since aggregation is largely connected with latency, the benefits from aggregation tend to be shared across the areas of faster scheduling, more scalability, and faster WAN operations.

11. Conclusion

Combining the techniques of multi-threaded processing of data and metadata with the concept of small file aggregation can result in significant performance increases. These increases can be further improved by adding techniques such as conditional updates or conditional file transfers. Performance increases above factors of 1000 have been observed. In addition, using user-generated aggregates can result in significant decreases in archival system metadata.

12. Performance Results

The following results were obtained on a cluster of 4 client nodes connected to to a Panasas file system,
For various files sizes and commands.

Local Mode

Mode	cp 1KB (files/s)	cp 10MB (MB/s)	cp 1GB (MB/s)	conditional transfer (files/s)	chmod (Files/s)	find (Files/s)	rm (Files/s)
sh	32	15	55	-	47	312	247
psi (local)	1,813	398	431	2364	2,090	1,743	2,999

Also, in a recent large scale test on 16 nodes connected to a Panasas file system, 295 TB of data (consisting of 967,000 files) were copied at an average rate of 2.85 GB/sec.

The following results were obtained on a cluster of 4 nodes, from a Panasas file system to Los Alamos HPSS.

Archive Mode (HPSS)

Mode	cp 1KB (Files/s)	cp 10MB (MB/s)	cp 1GB (MB/s)	cond (files/s)	chmod (Files/s)	find (Files/s)	rm (Files/s)
psi (HPSS)	80	1,071	580	102	295	599	155
psi (HPSS, TAR)	1,139	256	269	2,365	31,188	2,999	15,210

The following results were obtained on a cluster of 4 nodes, from a local Panasas file system to a remote Lustre file system, with a round trip time of 38 ms (Los Alamos, NM to Livermore, CA)

Host-to-Host Mode

Mode	cp 1KB (Files/s)	cp 10MB (MB/s)	cp 1GB (MB/s)	cond (files/s)	chmod (Files/s)	find (Files/s)	rm (Files/s)
psi (h2h)	1,933	423	480	2,396	2,433	3,012	229

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Richard Hedges
Lawrence Livermore National Laboratory
hedges1@llnl.gov

ABSTRACT / SUMMARY

This position paper discusses issues of usability of the large parallel file systems in the Livermore Computing Center. The primary uses of these file systems are for storage and access of data that is created during the course of a simulation running on an LC system.

INTRODUCTION

The Livermore Computing Center has multiple, globally mounted parallel file systems in each of its computing environments. The single biggest issue of file system usability that we have encountered through the years is to maintain continuous file system responsiveness. Given the back end storage hardware that our file systems are provisioned with, it is easily possible for a particularly I/O intensive application or one with particularly inefficiently coded I/O operations to bring the file system to an apparent halt.

The practice that we will be addressing is one of having an ability to indentify, diagnose, analyze and optimize the I/O quickly and effectively.

Tools applied

LMT: LMT[1] (Lustre monitoring tool) is run by the Livermore Computing system administration staff to monitor operation of the Lustre file systems. It is generally used to probe and isolate reported problems rather than to identify the problem before or as it develops.

Having an earlier version of LMT accessible to users proved problematic. The particular issue was that some users would "cry wolf" when they saw periods of heavy usage of a file system. These notifications were generally self serving and counter productive, so presently LMT is available for system administrators only.

In daily operations, Lustre system administrators may have running instances of LMT, but would not necessarily be tracking the output, unless a problem (such as a file system being sluggish or unresponsive) had been reported. Due to the architecture of Lustre, LMT leads one down an indirect path in identifying the source of a file system load. Load is observed on storage or metadata servers, next correlated with client activity, and finally (hopefully) identified with a single users job. This detective work can take some time, so it can be a challenge to do all of the tracing while the offending code instance is still active.

Darshan, strace: Since a single application program with inefficiently coded I/O operations can have center wide negative impact on parallel file system function and usability, It is critical to be able understand the sequence of I/O operations that a code is generating and to understand the effects of those on file system behavior.

For some time we have been in the business of profiling file system I/O for selected applications to diagnose and resolve performance problems causing center wide impact. Initially profile data was extracted exclusively from strace [2], and application runs and analyzed essentially by manually reviewing the data.

We generally trace with the options “strace -tt -etrace=file,read,write,close,lseek,ioctl” which provides time stamped system call traces to standard error for the I/O related system calls identified. We can collect the system call traces on a per process basis. It is possible to trace a running process, or to incorporate the tracing in a job run script.

More recently we also use Darshan[3] from Argonne National Laboratory. Darshan is a petascale I/O characterization tool. Darshan is designed to capture an accurate picture of application I/O behavior, including properties such as patterns of access within files, with minimum overhead. Darshan includes scripting to analyze and aggregate the data.

A code can be instrumented with Darshan by utilizing wrapper scripts, or by interposing the libraries using LD_PRELOAD. Being as lightweight as it is makes it suitable for full time deployment, although we at LC do not apply it in that manner.

These methods are available to users, but have primarily been applied by an LC staff member on behalf of a user or application team. Note that these methods are also applied in the case where the performance issues impact the user’s productivity, even if the center-wide impact is minimal.

User training and documentation

Training specific to application I/O performance issues is summarized in two documents maintained on the clusters in /usr/local/docs: (1) Lustre.basics and (2) Lustre.striping. We have also included I/O specific discussions in user oriented system status meetings on a regular basis. Consulting is available, and is offered on a general and on an intervention basis.

Let me interject a personal comment here related to user training, because I am eager to see if others at the workshop have observed similar. Relative to other parts of parts of a complex HPC system (e.g. processor architecture, code parallelization) I find that our user community seems generally more resistant to learning about the I/O architecture and how to use and code to it effectively. I suspect that this is a historically bias that the CPU processing is the valuable resource and the I/O bandwidth and storage capacity are free. We at the center may have reinforced this, if subtly, by our accounting and allocation policies.

CONCLUSIONS

For the key initial step of identifying a code or user who, by their I/O actions are impacting the user, we have a workable tool. LMT provides an path, albeit indirect to associate system load with a particular root cause.

The strace and Darshan offer approaches (some overlapping and some complementary) to analyze the I/O execution of an HPC application. They allow one to identify and localize a problem in a code.

We have an issue on the user training side. Some code teams have taken on the challenge of understanding good application I/O practices. Others have been motivated only when I/O performance was an insurmountable hurdle.

REFERENCES

1. LMT github site https://github.com/chaos/lmt/wiki
2. strace: stardard Linux command to “trace system calls and signals” see “man strace”
3. Darshan: Petascale I/O Characterization Tool *http://www.mcs.anl.gov/research/projects/darshan/*

U.S. Department of Energy Best Practices Workshop on
File Systems & Archives
San Francisco, CA
September 26-27, 2011
Position Paper

NCAR's Centralized Data Services Design

Pamela Gillman
NCAR Computational & Information Systems Lab
pjg@ucar.edu

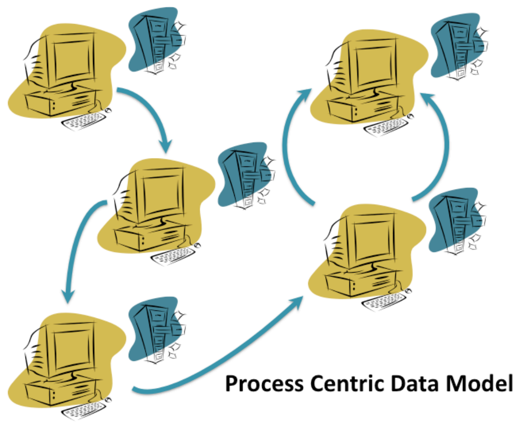
Erich Thanhardt
NCAR Computational & Information Systems Lab
erich@ucar.edu

ABSTRACT / SUMMARY

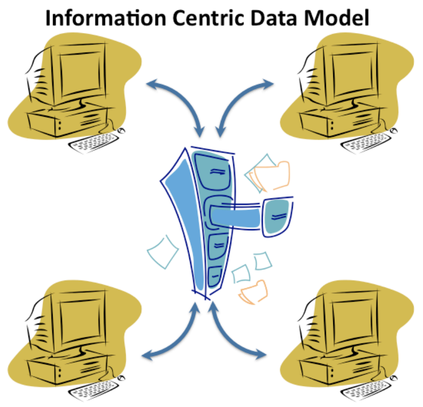
As NCAR designs and builds our next generation computational center, we are exploring ways to evolve scientific data workflows from a process centric model to a more information centric paradigm. By looking at cyberinfrastructure design, resource allocation policies and software methodologies, we can help accelerate scientific discoveries possible from computational resources of this scale. We will explore the challenges we have identified in our data architecture and present some of our current projects moving us towards an information centric solution.

INTRODUCTION

Computational centers have traditionally provided systems architected for a single use such as computation, data analysis or visualization. Each resource has local storage configured for the typical task performed and the center provides a common tape based archival system. This encourages a scientific workflow that typically involves retrieving input data from the archive system, generating new data, including intermediate files necessary for the next run, and finally storing all data back on the archive system. If you wish to post-process, analyze or visualize the data, you need to read the data back onto a different resource. This process may repeat multiple times as either issues are identified in the data, or discoveries spark a new direction of research.



When surveyed, users identified the movement of data between resources as a significant bottleneck in their workflow. Therefore, armed with this information, and driven by the ever-escalating costs of archival systems and the increases in ability to produce data, we started looking at architectural solutions to evolve workflow. The traditional workflow model is very process centric. Data moves between resources dedicated to a single step in the overall process and the archive essentially becomes a file server. What if we started looking at the data as the center of the workflow? Not only would this decrease the number of data movements in the workflow, but it can potential decrease the amount of data ultimately targeted for actual archiving. We now refer to this as an information centric model.



Evolving Scientific Data Workflow

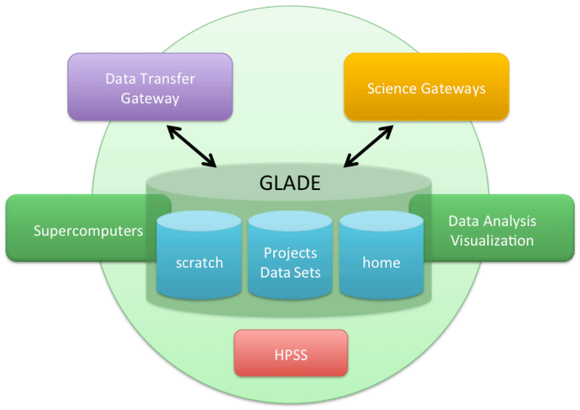
Based on an analysis of current workflows, we identified several challenges in evolving data workflow toward the new paradigm. Many bottlenecks exist in current workflows; it's time consuming to move data between systems; bandwidth to the archive system is insufficient; and available disk storage space is insufficient.

This presents a bit of a 'chicken and egg' problem. The current environment potentially shapes current user behavior. How do we anticipate the behavioral changes that will occur with a significant change in the environment? Storage cost curves are steeper than compute costs so how do we find the right balance between storage and compute investments? Archive costs are on an unsustainable growth curve so how do you better balance usage of disk space versus archive space?

Globally Accessible Data Environment

The first step we took was to centralize our data analysis and visualization resources around a centralized storage system. This was a lower bandwidth parallel file system solution that provided users of these resources a single namespace for their work. Space was provided not only for short-term 'scratch' usage but also for longer-term project spaces necessary for data analysis and visualization work.

The next step was designing a scalable architecture that encompassed all HPC resources including access to data collections managed by NCAR. This architecture needed to be flexible enough to support current systems and science gateways, and also able to scale as HPC resources grow with the new data center. And to ensure the shift in workflows, the user needs to be able to interface with this environment in the same way no matter which resource they are working from or what task they are trying to accomplish. High bandwidth connectivity to any resource within this environment and a choice of interfaces support current projects and are flexible enough to support future requirements.



The GLADE data architecture becomes the centerpiece of the new information centric workflow. Data can stay in place through the entire process as GLADE provides not just 'scratch' space for computational runs, but persistent longer-term storage for data processing, analysis and visualization. This persistent storage allows completion of the entire workflow prior to final storage of results either at NCAR or offsite. The addition of high-bandwidth data transfer services with direct access to the GLADE environment provides efficient methods for moving data between NCAR and peer institutions.

Accounting Systems Enhancements

To allow for better tracking of resource usage, the NCAR accounting system is being redesigned to account for computational resources, data analysis and visualization resources, disk storage usage, data transfer services and archival services. These tools will allow NCAR to fine tune the balance between resources based on evolving usage patterns due to changes in workflow.

Workflow Examples

Case 1: Nested Regional Climate Model (NRCM)

The project group has common access to 'scratch' space and a dedicated longer-term project space. The computational team submits a model run to the supercomputing queue. The model outputs approximately 100 variables per time step along with intermediate data files associated with startup of the next time step to the 'scratch' file system. Once the model completes, a post-processing job pulls approximately 20 variables of interest into data analysis files writing these into their project space. The analysis files are now available for analysis by the research team. This data will stay in place as long as necessary to complete the analysis. A final job step writes the final full output files to the HPSS archive. Since the 'scratch' file system is purged regularly, intermediate files that are no longer needed are never stored on the archive, and smaller data sets needed for analysis are available to the team right after the computational run completes.

Case 2: Research Data Archive (RDA)

The Research Data Archive provides access to common data sets to the research community. Prior to the implementation of the GLADE architecture this data was only available from the archive system. By allocating space within GLADE for the RDA data, access can now be granted directly from NCAR's HPC resources. Previous workflows needed to first copy this data from the archive to a 'scratch' area before running the computational job. There were costs in time required to access the data, space required to hold the data and the side effect of numerous copies of the same data being on disk at the same time. With direct access now available, jobs use the data from a central location that's immediately available to all jobs and doesn't rely on archival access..

CONCLUSIONS

We feel that we have made progress towards a better architecture to meet the diverse needs of our user community. We believe that this architecture is sustainable into the future and will help balance the costs associated with compute/storage/archival. Checks are in place so adjustments can be made as user behaviors change and hopefully data management becomes not just a tedious task, but also something that results in more productive scientific discovery.

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

Kevin Glass
PNNL
kevin.glass@pnnl.gov

ABSTRACT / SUMMARY

MyEMSL is a data management system for scientific data produced at EMSL. The data must be made accessible to users either by a simple directory-style search, a metadata search or as part of a workflow. To provide these features the system requires several points of interaction between users and the EMSL archive.

This position paper addresses the workshop's Usability of Storage Systems track and summarizes what we have accomplished in the development of MyEMSL and some of the challenges that remain with regard to the interaction between users and our archive.

INTRODUCTION

The MyEMSL data management system was developed to collect and archive data produced by EMSL instruments. The data is made available to the scientists who are allowed access to the data via the web and to analyze that data using EMSL's computational resources, including Chinook, EMSL's supercomputer.

EMSL boasts more then 150 scientific instruments that are capable of producing terabytes of data each week. The data collected by EMSL instruments comes in a wide variety of formats including images, spectrographs, single value outputs. The data will appear as files ranging from a single file on the order of 100 GB file to thousands of files on the order 1 MB each. Some of the instruments will produce only a single file, which currently is entered manually into a lab notebook.

In addition to getting EMSL users their data, the MyEMSL archive was designed to be a node in a workflow. Raw and processed data is stored in the archive, which is connected to computational resources. For example, the user can transfer data from the archive to a search node running Hadoop. The results of this search will be transferred back to the MyEMSL archive, then passed to Chinook for further processing and ultimately presented to the user through a visualization tool.

As with most of MyEMSL, this technique is in its infancy. We are examining options for critical parts of the infrastructure though most much of it is in place and working.

1. DESIGN OF MyEMSL

The design philosophy for MyEMSL was to let someone else do the work. The system relies on several open source software products such as Apache and SLURM. The goal was to select the optimal product for each type of component by testing the performance and ease of use for each product. However, budget, time and other factors limited the amount of testing and development available to the team.

Exacerbating this problem was the need to understand the use cases for hundreds of scientists using hundreds of different instruments. Prior to the development of MyEMSL, the development team reviewed information collected from previous attempts to develop data management systems for EMSL. These records included information regarding the amount of data collected by each instrument, the number and sizes of the files produced, the format or formats of the data and whether or not further processing of the data was required before it was released to the end user. The instrument information survey was originally collected on paper. It has since been moved to a web form and repopulated. As new instruments are added to EMSL's instrument suite, the person responsible for the instrument will complete this form.

In addition to collecting fundamental information regarding new equipment, the developers need to collect information regarding the metadata associated with the system. Given the number and specialization of each instrument, collecting the appropriate information from a one-on-one interview is infeasible given our time constraints. To address this problem, we are implementing two features in this system: a metadata form builder and metadata extractors to collect metadata that is automatically collected by the instrument and stored in the raw or processed data files.

2. COLLECTING METADATA

Given the evolving nature of science and scientific inquiry, the required metadata collection set will necessarily evolve. This requires MyEMSL to allow for a flexible metadata storage system. We define metadata to be data used by the end user for searching; a description of where the results of an experiment reside. When data is transferred from an instrument to the archive, the data is divided into metadata and raw or processed data. The raw or processed data is stored in the archive as a set of files and the metadata is stored in a database. To facilitate searches, MyEMSL generates specific databases based on the field of interest. These act as index caches for common—field based—searches. For example, searches for system biologists may include genetic information and ignore thermodynamic properties gathered from a single experiment. The main point is the scientist must be the person defining what is important for the scientist.

The specifics of metadata storage are still under investigation. We are currently considering two possibilities: a large relational database and a quad store. We are experimenting with these options using three criteria: performance, ease of implementation and ease of use. Of these options, only performance is an objective measure. The others are clearly subjective and will be tested within the constraints of our budget.

The management of raw, processed and metadata lead to two questions: how do we collect provenance data and how do we define data using common or standardized mechanisms? The collection of provenance data begins with an EMSL user submitting a sample for analysis. As this is the only person who can legitimately define the nature of the sample, the user is required to complete a "sample submission" form. Currently, this form collects only the basic data regarding a sample, such as the name of the submitter, the date of submission and so on. As with the collection of experiment metadata, the information required for a given sample type must be defined by the scientists responsible for operating the instrument.

Other sources of provenance data include the configuration of each instrument used to process a sample, the operating environment of the instrument and a description of the workflow used to process a sample. Some of the data is automatically collected from the instrument or detectors near the instrument. Others must be manually entered before data is loaded into the archive. This presents a user-interface problem. The system must minimize requirements for manual data collection and it must ensure the data is collected. We are still working on this problem.

3. UNIFYING DATA STANDARDS

The question regarding common or standardized data representations is a difficult one to answer. The scientific community agrees on few standardized representation and any level. Thus, there are several "standards" for each field and implementing all of them would be impossible. Our approach to this problem is to adopt a single, recognized standard for storing metadata and to use field-specific representations for the index caches. We are currently working with Nanotab [1] to represent both nanoparticle and some biological data. To generate the index caches, MyEMSL requires translators from Nanotab to other domain standards. We are currently investigating, with input from the scientific community, how to implement this feature.

The central feature of MyEMSL is its ability to move files of varying sizes to the archive and to retrieve those files. Before a scientist performs and experiment, he or she must configure a listener called ScopeSave. ScopeSave will monitor a specified directory and, when the experiment is complete, it will archive and compress the directory. The archived data files are sent to the archive via an Apache server to a storage queue managed by SLURM. When the archive is ready to store the input data, the data is transferred from the queue to the archive.

CONCLUSIONS

MyEMSL is a data management system currently in operation at EMSL however we are continuing to investigate some features. The system can take data from scientific instruments, load the raw, processed and metadata in a database and makes this data available to users. The main issues under investigation relate to the ease of use of the system by end users. These include features that are directly accessible by the user, for example, data transfer from an instrument to the archive and features that are indirectly accessible, such as the metadata database.

REFERENCES

1. caBIG Nanotechnology Working Group http://sites.google.com/site/cabignanowg/home

U.S. Department of Energy Best Practices Workshop on
File Systems & Archives
San Francisco, CA
September 26-27, 2011
Position Paper

Ruth Klundt
Sandia National Labs
rklundt@sandia.gov

Stephen Monk
Sandia National Labs
smonk@sandia.gov

John Noe
Sandia National Labs
jpnoe@sandia.gov

James Schutt
Sandia National Labs
jaschut@sandia.gov

ABSTRACT

We here present an overview of our current file system strategies, and a brief mention of planning for the future. The focus of the discussion is the link between usability issues and implementation decisions.

INTRODUCTION

Our primary drivers in the design of file system solutions are reliability and performance. In addition we attempt to provide solutions covering a spectrum of user needs, which also include convenience of use, backup capability, and high availability.

Overview of Current File Systems

User requirements in the storage arena are often difficult or impossible to satisfy simultaneously with a single global solution. Simulation codes generate a large quantity of restart data that must be stored quickly, as a defense against system outages. Most of this data is transitory, so does not need to be backed up. Other types of user data such as application codes and input data must be stored reliably. During periods of maintenance, it is important to users for the continuity of their work that some portions of the infrastructure remain available.

At present we maintain three basic categories of storage.

- Site-Wide Parallel File System

  Our parallel file system is implemented using Lustre [1] running on commodity servers, backed by DDN 9900/9550 raid cabinets. This file system serves ~2PB of fast scratch space to 4 different clusters, via LNET routers. Testing is under way on an upgrade to DDN SFA10K hardware providing ~3PB space for the new TLCC2 installation. Software support for Lustre is provided by Whamcloud [2].

- Intermediate NFS File System

  On all clusters, a large storage space is delivered by means of Sun Unified Storage (7410) using ZFS. This is not purged, and not backed up.

- Traditional NAS

  Less than 100TB, provided by NetApp hardware backed up to corporate archives. Stable, safe, slow location serving user /home and /projects space commonly across the clusters.

Usability Impact

The parallel file system satisfies the need for fast storage of large data sets. Although no backups can be done at this size, all possible efforts are made to avoid data loss, by means of hardware RAID configurations and continuity by means of Lustre failover and Multi-path IO. The local Red Sky Lustre implementation, which requires use of software RAID on the Sun equipment, has encountered some difficulties due to increased operational demands and is slated to be shutdown in favor of site file systems.

The intermediate NFS file system provides an alternate location for users to continue work during maintenance periods on the parallel file system. The longevity of the Sun 7410 platform is not clear given the lack of a clear hardware roadmap from Oracle. Although it has proven to be a solid product within this role, we are moving to a solution that is less of a "black box" from the view of the hardware (see below).

The NetApp filers serving /home and /projects have a fairly long history of providing robust reliable service here, although of limited size. New or different solutions have a high bar to meet in order to be considered as replacements for this functionality.

Future Plans

- GPFS NAS

  Some DDN 9550 cabinets are currently being re-purposed for use with IBM's GPFS file system [3] as an alternate highly available storage space, implemented at minimum cost. Production deployment is imminent.

- Ceph

  An effort is in progress to test the robustness, usability, and performance of the Ceph file system [4]. Early results show promise for this open source solution as a potential alternate in the NAS file system space in the near future. In addition, a variety of use cases other than HPC are being actively explored elsewhere, such as the ability to export as NFS, integration with PNFS [5], and access via user space clients. Interest in Ceph from disparate data storage venues can only improve the robustness of the implementation, and a broad user base provides some confidence that the file system has a productive future ahead.

  Some key design elements that make Ceph a high performance file system of interest:

  - Workload scalability (lots of servers/clients)
  - On-line expansion (easy to add capacity and performance)
  - Data replication (fault tolerance without RAID controllers)
  - Adaptive meta-data server (scalable)
  - Ability to reliably use commodity storage platforms

In conjunction with the Ceph testing effort, a heterogeneous test bed is being expanded and shared as a release test platform for production machines.

CONCLUSIONS

Challenges in maintaining multiple types of storage might be mitigated in the future, with improvements in current parallel file systems with respect to reliability and availability. Ideally a single global file system solution with pools of storage configured for different use cases would streamline the delivery of the disparate services needed. A single solution capable of providing sufficient bandwidth to parallel platforms, differential backup capabilities, and 24/7 availability to users does not yet exist.

REFERENCES

1. Lustre http://www.lustre.org
2. Whamcloud http://whamcloud.com
3. GPFS http://www-03.ibm.com/systems/software/gpfs/
4. Ceph File System http://ceph.newdream.net
5. PNFS http://www.pnfs.com

U.S. Department of Energy Best Practices Workshop on

File Systems & Archives

San Francisco, CA

September 26-27, 2011

Position Paper

doebpw5@nersc.gov.

Yutaka Ishikawa
University of Tokyo
ishikawa@is.s.u-tokyo.ac.jp

ABSTRACT

Two usability issues in storage systems connected with supercomputer are described. One is metada access and the other one comes from open source codes handling file I/O. In the latter one, because the users do not want to improve such codes, they request us to install faster disks such as SDD. According to our experiment, after improving the code, the performance is twice faster. Though twice faster disk was used, the performance was only 10 to 20 % gain. Another topic is related to a distributed shared file system being designed and deployed in Japan.

INTRODUCTION

Information Technology Center at University of Tokyo provides two supercomputer resources, SR11000 and HA8000 cluster, for domestic academic users. SR11000 is six years old machine and will be replaced with this October. HA8000 cluster consists of 952 nodes each of which has two AMD Opteron 8356s (16 cores). Each supercomputer connects with the proprietary parallel file system called HSFS. The total storage size is 1.5 PB.

In addition of computational resource services, we are currently designing and deploying distributed shared storage system, whose total size will be more than 100 PB, accessed by Japanese supercomputers including K computer, as the nation-wide high performance computing infrastructure. This infrastructure is called HPCI (High Performance Computing Infrastructure) supported by Ministry of Education, Culture, Sports, Science, and Technology. As shown in Figure 1, there two storage HUB, West and East. In West HUB, 10 PB storage and 60 PB tape archive will be deployed with K computer. 12 PB storage and 20 PB storage will be deployed in our university. The system is currently under construction and it will be operated from fall in 2012.

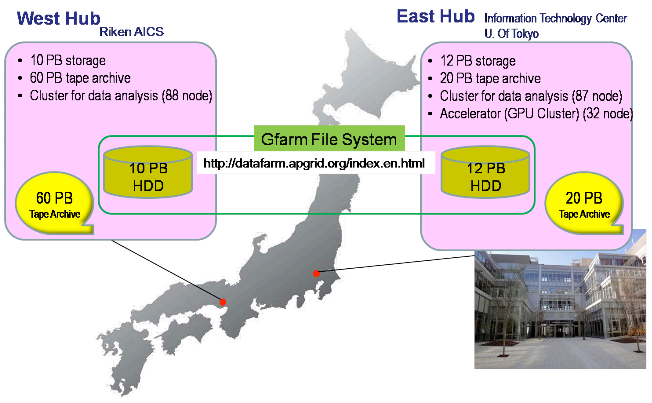

West Hub
Riken AICS
10 PB storage
60 PB tape archive
Cluster for data analysis (88 node)
East Hub
Information Technology Center
U. Of Tokyo
12 PB storage
20 PB tape archive
Cluster for data analysis (87 node)
Accelerator (GPU Cluster) (32 node)
Gfarm File System
http://datafarm.apgrid.org/index.en.html
10 PB HDD
12 PB HDD
60 PB Tape Archive
20 PB Tape Archive

Figure 1 HPCI Storage

Usability Issues in HA8000 cluster

As many others pointed out, the interactive users complained slow meta data accesses in the HSFS parallel file system especially "ls –l" command that involves many meta data accesses to obtain all file statuses. To provide faster response for the interactive users, the vendor has modified its system to handle the interactive users' requests first. This modification with other several changes mitigates this slowness.

Another issue comes from open source codes that are not well programmed for file I/O. The users believe that low performance of such a code results in slow file I/O access, but this is sometimes not true. For example, a bio informatics tool, used by our bio informatics users, consists of two processing modules, genome alignment and data format change. In the genome alignment processing, there are so many critical regions, and the low performance results in those regions. After reducing the number of critical regions, the program is twice faster. In data format change processing, it opens and reads the same file about 1000 times and eventually reads 1 TB data in total. To eliminate this silly code, the program is twice faster. The users have thought if the file system is twice faster, the program would run twice faster. But, though the file system is twice faster using SDD, the performance is only 10 to 20 % improvement.

Usability Issues in HPCI storage system

This workshop may not consider distributed shared storage systems shared by different organizations. But we would like to address this kind of storage systems because data-sharing is important in the data-intensive science. One good example is ILDG (International Lattice Data Grid) where lattice QCD (Quantum chromodynamics) data generated by supercomputers are shared by international organizations. Another example is the climate simulation field. As far as we understand, the research group develops their simulation code, obtains data generated by the simulator, and after the generated data are examined and new information for them is obtained, the data are open to others who are interested in data for other purposes. Thus, the data is eventually shared by others.

The Japan HPCI tends to provide storage resource for not only traditional computational sciences but also data-intensive sciences including life science/drug manufacture, new material/energy creation, global change prediction for disaster prevention/mitigation, manufacturing technology, the origin of matters and the universe. To provide better usability for those users, issues are listed below. All issues arise because many research fields use the storage system, it is not yet predicted that their peek and sustained demands.

- ✓ Prediction of storage capacity
- ✓ Prediction of amount of file transfers

CONCLUSIONS

Dusty code must be replaced with modern code to provide usability for such application users. We have to much pay attention of distributed shared file systems with local file systems.

Appendix B: Workshop Agenda

5th Best Practices Workshop for High-Performance Center Managers
San Francisco, CA
Marriott Marquis Hotel
September 26-27, 2011

Monday, September 26

7:30-8:30	Breakfast and registration
8:30-8:45	Welcome (Club Room) *Jason Hick, Lawrence Berkeley National Laboratory, and Yukiko Sekine, U.S. Department of Energy, SC*
8:45-9:00	Thuc Hoang, U.S. Department of Energy, NNSA
9:00-9:15	Yukiko Sekine, U.S. Department of Energy, SC
9:15-9:30	Instructions for breakout sessions
9:30-10:00	Breakout Sessions ➢ Business of Storage Systems (Club Room) ➢ Administration of Storage Systems (Foothill B) ➢ Reliability and Availability of Storage Systems (Foothill D) ➢ Usability of Storage Systems (Foothill E)
10:00-10:30	Morning break
10:30-12:00	Breakout Sessions Continued
12:00-1:00	Lunch (Foothill G)
1:00-3:00	Breakout sessions continue
3:00-3:30	Afternoon break
3:30-4:00	Breakout sessions continue
4:00-4:15	Business Breakout: collection of thoughts/outbrief to entire group
4:15-4:30	Administration Breakout: collection of thoughts/outbrief to entire group
4:30-4:45	Reliability Breakout: collection of thoughts/outbrief to entire group
4:45-5:00	Usability Breakout: collection of thoughts/outbrief to entire group

<u>Tuesday, September 27</u>

7:30-8:30	Breakfast
8:30-9:00	Checkpoint and directions to breakout leaders
9:00-10:00	Breakouts continue
10:00-10:30	Morning break
10:30-12:00	Breakouts continue
12:00-1:00	Lunch
1:00-2:00	Breakouts continue
2:00-2:30	Business Breakout: collection of thoughts/outbrief to entire group
2:30-3:00	Administration Breakout: collection of thoughts/outbrief to entire group
3:00-3:30	Afternoon break
3:30-4:00	Reliability Breakout: collection of thoughts/outbrief to entire group
4:00-4:30	Usability Breakout: collection of thoughts/outbrief to entire group
4:30-5:30	Plenary workshop summary and next steps (report)

Breakout Sessions

The Business of Storage Systems (Club Room)

Sarp Oral, Oak Ridge National Laboratory and
David Cowley, Pacific Northwest National Laboratory

After reviewing position papers, here are the topics of discussion towards identifying best practices:

- Combining in-house expertise with open source and proprietary solutions and managing the vendor relationship
- Using COTS in HPC storage
- Planning and implementing center-wide file systems
- Establishing I/O requirements
- Dealing with exponential data growth
- Evaluating and integrating new technologies
- Making effective use of storage hierarchies

The Administration of Storage Systems (Foothill B)

Susan Coghlan, Argonne National Laboratory and
Jerry Shoopman, Lawrence Livermore National Laboratory

Plan

We have selected 5 position papers and a single topic from each paper. We are asking the authors to prepare 2-3 slides on that topic from the paper. During the breakout session, for each paper, an author will present their slides and the group will discuss. We chose to do it this way because most of the papers had a lot of topics and we didn't feel there would be time for each author to present all of them.

Administration Breakout Agenda

1. Review plan for the day, scope of discussion, get feedback for modifications, finalize plan (15 mins)
2. Position paper presentations and discussion (1.5 hrs - {10 mins presentation, 10 mins discussion} x 5 position papers)
3. Free discussion (30 mins)
4. Finish preparing report (30 mins)

Topics/Papers

1. Tiered solutions (Performance improvements) - Torres/Scott paper [LANL]
2. Data integrity/Availability - Heer paper [LLNL]
3. Disk quotas (managing growth) - Cardo paper [NERSC/LBNL]
4. Performance over time - Harms paper [ALCF/Argonne]
5. Configuration management and change control - Hill/Thach paper [OLCF/ORNL]

The Reliability and Availability of Storage Systems (Foothill D)

Mark Gary, Lawrence Livermore National Laboratory and
Jim Rogers, Oak Ridge National Laboratory

- Resilient and fault tolerance - RAID, backup, multi-path
- Data integrity - checksums
- Daily operations - software maintenance
- Off-hour support and availablity
- Monitoring and tools
- Other - leveraging procurements, contractual reliability commitments

The Usability of Storage Systems (Foothill E)

Shane Canon, Lawrence Berkeley National Laboratory and
John Noe, Sandia National Laboratories

Scope of the session and boundaries

- Between usability and administration
- Between usability and data management and data formats
- Position paper authors present (slides or talk) also a few others with contributions. (1st hour)
- Free form discussion after lunch to pull forth ideas
- Last half hour to firm

Appendix C: Workshop Attendees

John Bent, Los Alamos National Laboratory
Ryan Braby, National Institute for Computational Sciences
Jeff Broughton, National Energy Research Scientific Computer Center
Shane Canon, National Energy Research Scientific Computing Center
Nicholas Cardo, Lawrence Berkeley National Laboratory
Geoff Cleary, Lawrence Livermore National Laboratory
Susan Coghlan, Argonne National Laboratory
Roger Combs, HPC Navy Program
David Cowley, Pacific Northwest National Laboratory
Kim Cupps, Lawrence Livermore National Laboratory
Philippe Deniel, Center for Atomic Energy, Military Applications Division
Mike Farias, Sabre Systems
Evan Felix, Pacific Northwest National Laboratory
Naoyuki Fujita, Japan Aerospace Exploration Agency
Mark Gary, Lawrence Livermore National Laboratory
John Gebhardt, AFRL DSRC
Pam Gillman, National Center for Atmospheric Research
Kevin Glass, Pacific Northwest National Laboratory
Stefano Gorini, Swiss National Supercomputing Centre
Stephan Graf, Forschungszentrum Jülich GmbH/Jülich Supercomputing Centre
Gary Grider, Los Alamos National Laboratory POC
Kevin Harms, Argonne National Laboratory
Richard Hedges, Lawrence Livermore National Laboratory
Todd Heer, Lawrence Livermore National Laboratory
Cray Henry, High Performance Computing Modernization Program
Jason Hick, National Energy Research Scientific Computer Center
Jason Hill, Oak Ridge National Laboratory
Thuc Hoang, U.S. Department of Energy, National Nuclear Security Agency
John Hules, Lawrence Berkeley National Laboratory
Wayne Hurlbert, National Energy Research Scientific Computer Center
Yutaka Ishikawa, The University of Tokyo
M'hamed Jebbanema, Los Alamos National Laboratory
Thomas Kendall, U.S. Army Research Laboratory
Dries Kimpe, Argonne National Laboratory
Ruth Klundt, Sandia National Laboratory
Rei Lee, Lawrence Berkeley National Laboratory
John Noe, Sandia National Laboratory
Lucy Nowell, U.S. Department of Energy, Advanced Scientific Computing Research
Jack O'Connell, Argonne National Laboratory
Sarp Oral, Oak Ridge National Laboratory
Alex Parga, National Center for Supercomputing Applications
Norbert Podhorszki, Oak Ridge National Laboratory
Mark Roberts, Atomic Weapons Establishment
James Rogers, Oak Ridge National Laboratory
Mark Roschke, Los Alamos National Laboratory
Tim Scott, Northrop Grumman

Yukiko Sekine, U.S. Department of Energy, Office of Science
Jerry Shoopman, Lawrence Livermore National Laboratory
Mike Showerman, National Center for Supercomputing Applications, Innovative Systems Lab
Kazuhiro Someya, Japan Aerospace Exploration Agency
Bert Still, Lawrence Livermore National Laboratory
Osamu Tatebe, The University of Tokyo
Kevin Thach, Oak Ridge National Laboratory
Erich Thanhardt, National Center for Atmospheric Research
William Thigpen, National Aeronautic and Space Administration, Ames
Aaron Torres, Los Alamos National Laboratory
Dominik Ulmer, Swiss National Supercomputing Centre
Andrew Uselton, Lawrence Berkeley National Laboratory
Dick Watson, Lawrence Livermore National Laboratory
Charlie Whitehead, Massachusetts Institute of Technology, Lincoln Lab
Yushu Yao, Lawrence Berkeley National Laboratory