Comparative I/O Workload Characterization of Two Leadership Class Storage Clusters

Raghul Gunasekaran, Sarp Oral, Jason Hill, Ross Miller, Feiyi Wang, Dustin Leverman
Oak Ridge Leadership Compute Facility, Oak Ridge National Laboratory
{gunasekaranr,oralhs,hilljj,rgmiller,fwang2,leverman}@ornl.gov

Abstract

The Oak Ridge Leadership Computing Facility (OLCF) is a leader in large-scale parallel file system development, design, deployment and continuous operation. For the last decade, the OLCF has designed and deployed two large center-wide parallel file systems. The first instantiation, Spider 1, served the Jaguar supercomputer and its predecessor, Spider 2, now serves the Titan supercomputer, among many other OLCF computational resources. The OLCF has been rigorously collecting file and storage system statistics from these Spider systems since their transition to production state.

In this paper we present the collected I/O workload statistics from the Spider 2 system and compare it to the Spider 1 data. Our analysis show that the Spider 2 workload is more more write-heavy I/O compared to Spider 1 (75% vs. 60%, respectively). The data also show the OLCF storage policies such as periodic purges are effectively managing the capacity resource of Spider 2. Furthermore, due to improvements in tdm_multipath and ib_srp software, we are utilizing the Spider 2 system bandwidth and latency resources more effectively. The Spider 2 bandwidth usage statistics shows that our system is working within the design specifications. However, it is also evident that our scientific applications can be more effectively served by a burst buffer storage layer. All the data has been collected by monitoring tools developed for the Spider ecosystem. We believe the observed data set and insights will help us better design the next-generation Spider file and storage system. It will also be helpful to the larger community for building more effective large-scale file and storage systems.

1. INTRODUCTION

As a U.S. Department of Energy (DOE) leadership computing facility, the Oak Ridge Leadership Facility (OLCF) at the Oak Ridge National Laboratory (ORNL) provides world-leading computing capabilities to scientists and engineers for accelerating major scientific discoveries and engineering breakthroughs for humanity, in partnership with the computational science community. The OLCF has deployed different HPC architectures that are 10 to 100 times more powerful than typical research computing systems. The OLCF currently hosts Titan, the world's second fastest supercomputer [1], as well as a number of analysis and visualization platforms. The OLCF is a leader in large-scale parallel file system development, design, deployment, and continuous operation. In the last decade, the OLCF has designed and deployed two large-scale center-wide parallel file systems, namely, the Spider 1 and Spider 2 systems [14, 9].

An understanding of current storage system workload is critical for architecting next generation systems, and in the design and development of storage subsystems. A number of characterization studies have been done at the storage server level [15, 5]; characterizing disk level traces [12], and a number studies on enterprise I/O workloads [4, 15]. There have been very limited studies on the characterization of HPC storage system workloads with trace data collected from the backend storage controllers. However, a number of HPC workload characterization studies have focused on I/O characterization from a scientific application perspective by instrumenting application code for I/O trace collection [2, 13]. Nevertheless, these studies have proven useful in understanding the application workloads for architecting storage systems. Our study is unique in a way that we are using the traces collected at the backend RAID controller to understand scientific user workloads and for provisioning storage system resources. Our prior work [6] was on the characterization and study of the workloads on our first center-wide parallel file system, Spider 1. The lessons learned from Spider 1 helped us in the design and deployment of our next file system, Spider 2. In this paper, we present a comparative study of the workload characteristics of our storage system as we scaled from Spider 1, a 240 GB/s 10 Petabyte, to Spider 2, a +1 TB/s, 32 Petabyte storage system serving over 20,000 filesystem clients at the OLCF.

We collected and analyzed file and storage system data from a large-scale production scientific supercomputing complex. Our data and analysis provides useful insights on the I/O patterns, in particular I/O bandwidth utilization, request size distributions, and request latency distribution. Our observation of drastically increased peak performance overshadowed by the relative low utilization echoes the design trend of employing a burst-buffer layer. Our analysis, though not exhaustive, can be beneficial to the larger community and can be used in building more efficient next-generation large-scale storage systems.

PDSW2015, November 15-20, 2015, Austin, TX, USA

ACM ISBN 978-1-4503-4008-3/15/11…$15.00
DOI: http://dx.doi.org/10.1145/2834976.2834985

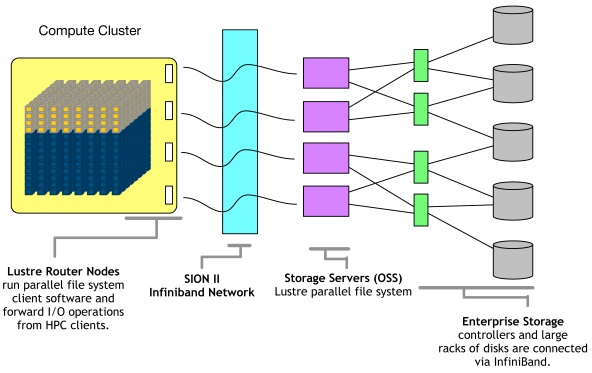


Figure 1: Spider Architecture

	Spider 1	Spider 2
Bandwidth	240 GB/s	+1 TB/s
Capacity	10 PB	32 PB
RAID Controllers	DDN S2A9900	DDN SFA12KX
Disk type	SATA	Near-line SAS
Number of disks	13,440	20,160
Disk redundancy	RAID 6 (8+2)	RAID 6 (8+2)
Number of OSTs	1,344	2,016
Number of OSSs	192	288
Lustre version	1.8	2.5
Connectivity	IB DDR	IB FDR

Table 1: Spider 1 and Spider 2 key specifications

2. SYSTEM OVERVIEW

Jaguar, a 1.75 petaflop system, was the flagship compute platform at OLCF between 2009 and 2013, and Spider 1 served the I/O needs of Jaguar. As OLCF was deploying Titan, a GPU accelerated 20-petaflop system in 2013, Spider 2 was deployed. The Spider storage system was scaled up in bandwidth and capacity to meet the needs of Titan. Table 1 highlights key characteristics of the Spider 1 and 2 file systems.

Figure 1 shows a conceptual architecture for Spider 2, very similar to Spider 1. The primary building block is a DataDirect Networks SFA12KX RAID controller pair (couplet) with 10 disk enclosures containing a total of 560 2TB NL-SAS disk drives. Eight Lustre OSS hosts are external to the SFA12KX and are connected to the couplet via 2 FDR Infiniband links on each host for path redundancy. In total there are 36 building blocks that are build into two file system namespaces on non-overlapping hardware resources (i.e., “atlas1” and “atlas2”). The resulting system was rated for, and tested to deliver an aggregate performance of 1.4 TB/s for reads and 1.2 TB/s writes which translates into 1+ TB/s aggregate read and write performance at the file system level.

Through the deployments of Spider 1 and Spider 2 the OLCF has built comprehensive monitoring tools[8]. Many of these monitoring tools have been in-house developed at the OLCF, particularly the “DDNTool”, a tool for monitoring the I/O requests from the DDN controller level.

DDNTool [8] was developed to monitor the DDN S2A and SFA storage system RAID controllers. Since the two DDN architectures have very different monitoring API’s, there are two programs: DDNTool for the S2A architecture and DDNTool_v2 for the SFA architecture. The two tools are very different in their implementations - DDNTool is a C++ program which communicates with the disk controllers directly over TCP/IP while DDNTool_v2 is a Python program which interfaces with a vendor-supplied python library which handles the low level communication - but they both accomplish the same basic task. The tools poll each controller for various pieces of information (e.g. I/O request sizes, write and read bandwidths) at regular intervals and writes this information to an in-memory MySQL database. At each poll interval, the old data in the database is overwritten with new data. By storing the data in a database, the data is available to numerous clients via a well-documented API. This allows multiple users to search and query in real-time.

For analysis purposes and understanding long term I/O trends, the data from the in-memory database is archived in a MySQL database. We looked at the bandwidth, transfer bytes and request size data during the study period of over a 4 months.

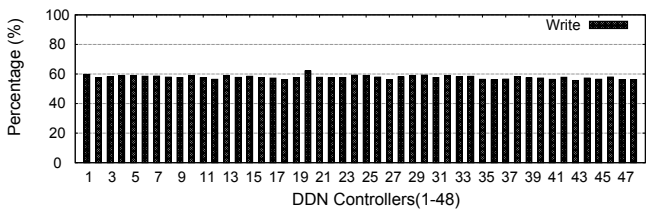
3. WORKLOAD CHARACTERIZATION

As a follow-up to our characterization of the Spider 1 storage system, below we will be presenting a comparative analysis of the I/O workloads observed on the Spider systems. Our analysis focuses on I/O usage trends and I/O request characteristics.

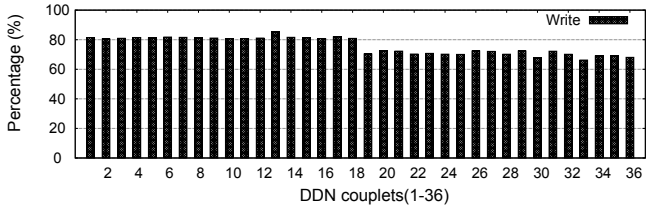
3.1 I/O Usage Trends

3.1.1 Read vs. Write Operations

Figure 2 shows the read vs. write balance observed on the two Spider systems. As can be seen, there is a slight imbalance in the total write requests for 50% of the controllers sampled in the Spider 2 system. The imbalance is because the upper level file systems are imbalanced. As we stated above, we deployed two file system namespaces on non-overlapping hardware resources. The Spider 2 file systems are allocated across projects at the OLCF and some projects have been heavier I/O consumers than others, and we have observed that some projects are transferring large datasets to the OLCF for processing on Titan. It turns out the “atlas1” partition is being utilized at a higher rate. In general we see a higher percentage of write operations on Spider 2 as compared to Spider 1, meeting our design expectations for handling large memory checkpoints with minimal reading of those checkpoint data files. Also, it is quite visible in Figure 2(b) that “atlas1” portion of Spider 2 is 10% more write-intensive than the Atlas2 (left half vs. the right half of the Figure 2(b). The discrepancy is due to the same factors and since being corrected the trend is moving back toward the expected balance. Finally, as can be seen in Figure 2, approximately 60% of the I/O workload on Spider 1 was write operations and 40% was reads. Spider 1 was a center-wide resource shared across all OLCF platforms and the high percentage of the read requests were attributed to analysis and data transfer I/O workloads accessing Spider 1. Spider 2 is also a center-wide shared resource, but it has a higher percentage of write operations than Spider 1 (approximately 75% vs. approximately 60%). We have conclude the way the system is being used accounts for the

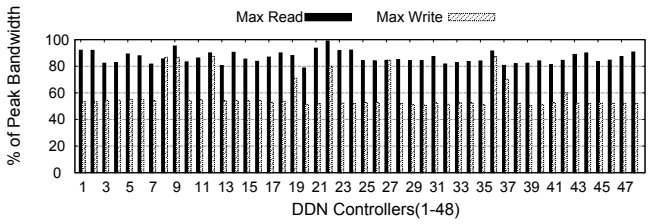


(a) Spider 1

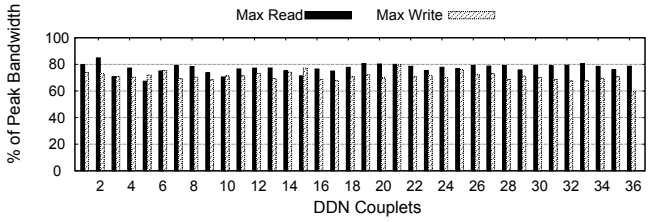


(b) Spider 2

Figure 2: Read vs Write on the Storage System



(a) Spider 1



(b) Spider 2

Figure 3: Peak bandwidth usage at the RAID controllers

differences; namely the Titan/Spider 2 users have more of a checkpoint/restart workload than on Jaguar/Spider 1.

3.1.2 Peak Bandwidth Usage

Figures 3(a) and 3(b) show the peak read and write bandwidth observed at the DDN RAID controllers for Spider 1 and Spider 2, respectively. Figure 3(b) shows the peak read bandwidth observed was 90% of the maximum theoretical performance for Spider 1 (3 GB/s per controller); we observed 80% of the maximum bandwidth for Spider 2 (36 GB/s per controller). For writes, the bandwidth observed was 50% for Spider 1 and 70% for Spider 2. These values show us our current read I/O workloads are achieving a slightly lower peak value of the overall percentage. The good news is the write operations on Spider 2 are obtaining a higher percentage of the peak performance. This can be attributed to better formed write I/O being issued from Titan. While the exact reasons are under investigation, we believe the middleware libraries such as ADIOS [7] are helping to improve the I/O characteristics.

3.1.3 Spider 2 Usage Stats

Figure 4, shows the aggregate bandwidth (i.e., the sum of read and write bandwidth) observed on Spider 2. We do not have a comparable data set for Spider 1, however, we think it is useful to share the Spider 2 data to further gain insight into the behavior of applications at this scale.

The Spider 2 file system was designed to deliver greater than 1 TB/s at the file system level if exercised with well-formed, large, sequential I/O on a quiet system. However, under normal production workloads, the file system is exercised by random mixed I/O workloads. Our early system tests showed a single NL-SAS disk drive can achieve 20-25% of its peak performance under random I/O workloads (with 1 MB I/O block sizes). Extrapolating from here, the expected Spider 2 performance should be at most 250 GB/s under such bursty production workloads. [10].

As can be seen from the Figure 4, more than 95% of the observed bandwidth was less than 50 GB/s. Around 4% of the time the bandwidth was over 50 GB/s. For less than 1% of the time the bandwidth peaked over 250 GB/s. These values are in line with the design goals of Spider 2, as typical scientific applications are compute intensive and perform I/O in bursts totaling less than 5% of their compute time allocations. This suggests these mixed workloads could benefit from a burst buffer storage layer to absorb and align their I/O.

Figure 5 shows the Spider 2 space utilization. As can be seen, more than 50% of the capacity is currently being used. For a sustained a bandwidth of 250 GB/s (peak random production bandwidth) for 10 minutes in a day, 146 TBs of data will be generated. Given the 32 PB aggregate capacity of Spider 2, this will consume all space on Spider 2 in less than a year. To prevent this, OLCF employs a "purge" policy. Periodically, files with a timestamp older than 14 days are deleted from the Spider 2 scratch file system. This helps maintain a utilization less than 75% on the Spider 2 system as we have performance degradation when file system utilization is higher. This performance degradation is due to longer seek times to reach free blocks on the disk drives in the RAID set; purging allows us to try to keep this seeking to a minimum at all times.

3.2 I/O Requests

3.2.1 Request Size distribution

Figures 6 and 7 show the distribution of read and write requests on Spider 1 and Spider 2, respectively. As can be seen, there are differences. The Lustre file system supports a range of request block sizes, the smallest being 4 kilobytes (kB) and the largest being 4 Megabytes (MB). However, on Spider 1, the smallest block size we were able to monitor was 16KB, a limitation of the DDN RAID controller. Interpreting from the CDF and PDF plots:

- 60% of write requests on Spider 2 are 4kB or less. In Spider 1, we were unable to distinguish between the 4kB, 8kB and 16kB sizes, which accounted for 50% of the write requests. Accounting for this and comparing Spider 1 and 2 file size distributions, it can be seen there are 10% more small block requests (i.e., smaller

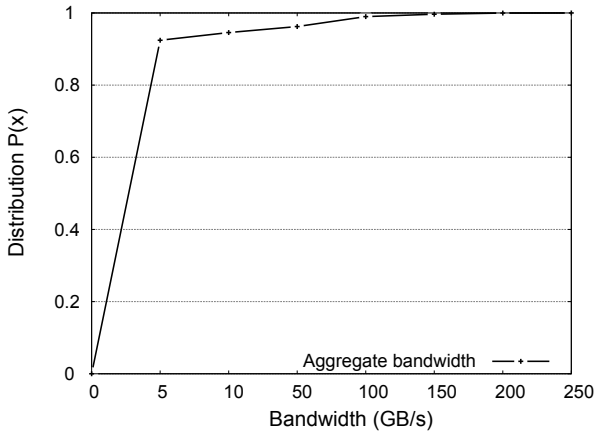


Figure 4: Cumulative Distribution Function (CDF) of Spider 2 usage with respect to bandwidth

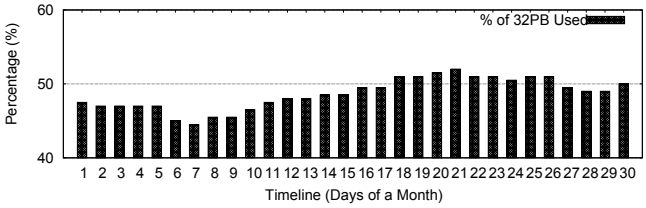
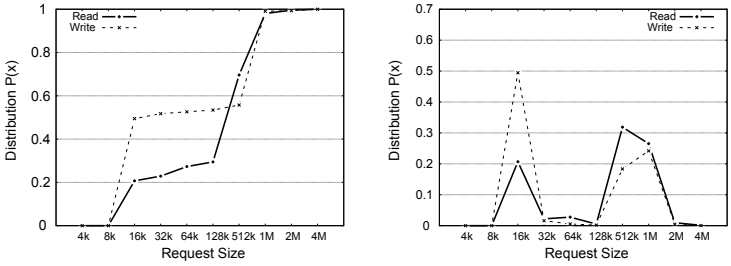
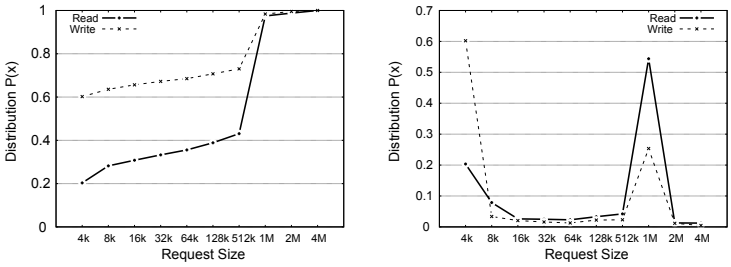


Figure 5: Spider 2 usage with respect to storage space



(a) CDF (b) PDF

Figure 6: Spider 1 - Distribution of Request Sizes



(a) CDF (b) PDF

Figure 7: Spider 2 - Distribution of Request Sizes

than 8kB) on Spider 2 for write operations. Please remember these statistics are directly obtained from the DDN controllers not from the file system. One possible explanation for this increase could be an increase in the local file system (ldiskfs) metadata operations. Another possible explanation could be an increase in the number of controller-level background disk integrity check (i.e., scrubbing) events. The exact cause of the increase is unknown at this point and it is being investigated. For read operations, Spider 1 and Spider 2 behave the same for small files; the distribution has not changed.

- For writes, 70% of the requests on Spider 2 are less than or equal to 512kB, whereas on Spider 1 only 55% of the requests were less than or equal to 512kB. It is worth noting the data presented in this study are gathered in different years, the application versions have changed, and there was a different mix of applications running as the data was collected. The OLCF traditionally has a large mix of applications which have well-formed I/O or are using middleware libraries such as ADIOS [7]. The latest round of applications do not have this robustness in their I/O patterns, and have shown to have some pathological tendencies. These smaller files and request sizes are more focused on read operations are large performance drains on a system designed to perform 1MB sequential write and read I/O operations.

- On Spider 2 over 50% of reads were 1MB, similar to Spider 1 combined 512kB and 1MB were 50%. However, only 25% of writes on Spider 2 are 1MB, whereas on Spider 1 over 45% of writes where either 512kB or 1MB. We postulate again that the application workload of the OLCF is different enough at the times of measurement to show this difference in the distribution.

- On Spider 1, we observed a large number of 512kB requests for both read and write requests. This was due to problems in the `dm-multipath` [11] package available for the version of the Linux operating system that was employed. This version had a bug that broke up a 1024kB I/O request into 2 512kB requests. Later versions addressed this performance problem by not breaking up the I/O request. Additionally Spider 1 used the `deadline` I/O request scheduler allowing the kernel to re-order some I/O requests. In 2011, Spider 1 was moved to the `noop` scheduler, and the phenomenon disappeared. We see the same drop in 512kB requests for Spider 2 as well. Finally, changes to the `ib_srp` kernel module allow more queued requests and better memory handling of a larger queue through scattergather tables, allowing fully queuing 1024kB requests and pushing them all the way to the DDN controller. As a result of these 3 changes the number of 512kB requests have been dramatically reduced for Spider 2.

3.2.2 *Request Size Latency distribution*

With the new DDN SFA API, we now have the capability to collect I/O request service latencies. Figure 8 shows Spider 2 request service latency distributions. Service latency in Figure 8 includes the total sum of the time spent

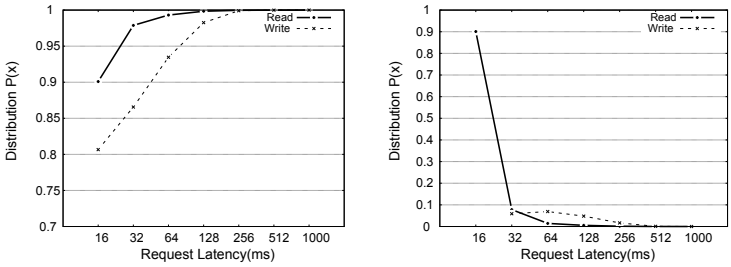
on the queue and the time spent on fetching/putting data from/to the disk. 90% of reads and more than 80% write requests were serviced in 16ms or less, and this is the finest granularity the DDN API can provide.

The SFA12KX's software incorporates newer caching algorithms that are designed to lower the latency for certain operations, and with the readahead cache disabled and Real-time Adaptive Cache Technology (ReACT^{TM})[3] enabled, we see this highly desirable distribution of latencies of 16ms or less.

- The SFA12KX disk controller has the ability to analyze incoming I/O request stream and begin to aggressively read-ahead on the disk to internal cache. The hope is a large read can be sped up by hiding the latency of the seek if the data is in cache. In our experience and testing, this was a very large hindrance to performance in workloads that mimic the many file read operations from Titan's compute nodes. This is due to the algorithm caching the wrong data, having to invalidate the cache, seek to the location on disk, cache that data, and then repeat with the next read request. By disabling the read-ahead cache, we dramatically lower the latency of the non-sequential read requests we see in the production environment.
- The Real-time Adaptive Cache Technology (ReACT^{TM}) feature was developed to make the SFA platforms more attractive from an IOPS perspective. The feature's main goal is to speed up the performance of the large 1MB write requests, avoiding caching the I/O on the partner controller. The decision to only cache <1MB write operations relieves a bottleneck that earlier versions of DDN's products did poorly - including the S2A 9900's that Spider 1 was constructed of. In earlier versions of the product the cache could either be mirrored on the partner controller for data resilience in the face of controller failure, or could be disabled to obtain the performance for large streaming 1MB (or larger) writes. ReACT gives a framework where the best of both worlds can exist. A 1MB block can be written directly to disk, which is the lowest latency possible for that operation. A 4kB block can be written into controller cache and also mirrored to the partner controller in about the same amount of time, which is far faster than committing that operation to the disk. We feel that this feature has allowed Spider 2 to reach the low latencies across the spectrum of write request sizes and is a large benefit to users.

4. CONCLUSIONS

Large-scale file systems are complex to architect and deploy and are equally complex to operate. Key to successful deployment and operation is understanding the I/O workloads exercising these file systems. At the OLCF we continue to pay particular attention to collecting file system statistics for better understanding how our file systems are used. Over the 5-year lifetime of Spider 1, we collected detailed operational data and we are doing the same with the Spider 2. Gathering, curating, and querying this data was essential in architecting the Spider 2 system, and we are expecting this effort will again be instrumental in building the next-generation parallel file system at the OLCF.



(a) CDF (b) PDF

Figure 8: Spider 2 - Distribution of Request Service Time(latency)

As the Spider 2 data shows us, we observed roughly 75% writes on the Spider 2 storage system. This is slightly different than what we have observed on Spider 1 and we conclude that Spider 2 is currently exercised with more write-heavy I/O workloads. At the same time our write requests consist of only 4kB and 1MB write requests. This is a good step in the right direction, since the 512kB write requests have been eliminated from the system due to the fixes in dm-multipath, I/O request scheduling and ib_srp kernel module feature development and bug-fixing. The existence of extra 512kB was detrimental to the overall aggregate I/O performance on Spider 1 and this has been corrected in Spider 2. Observing data we also now firmly believe a Burst Buffer layer between our compute platforms and the parallel file system will better serve our scientific workloads by absorbing and aligning the bursty, random I/O patterns and improving the overall application I/O performance.

Acknowledgment

This work was supported by the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is managed by UT Battelle, LLC for the U.S. DOE (under the contract No. DE-AC05-00OR22725).

5. REFERENCES

[1] A. S. Bland, J. C. Wells, O. E. Messer, O. R. Hernandez, and J. H. Rogers. Titan: Early Experience with the Cray XK6 at Oak Ridge National Laboratory. In *Proceedings of Cray User Group Conference (CUG 2012)*, 2012.

[2] P. Carns, K. Harms, W. Allcock, C. Bacon, S. Lang, R. Latham, and R. Ross. Understanding and improving computational science storage access through continuous characterization. *Trans. Storage*, 7(3):8:1–8:26, Oct. 2011.

[3] DataDirect Networks. SFAOS ReAct Cache. http://www.ddn.com/pdfs/SFA12KX_Whitepaper.pdf.

[4] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper. Workload analysis and demand prediction of enterprise data center applications. In *Workload Characterization, 2007. IISWC 2007. IEEE 10th International Symposium on*, pages 171–180. IEEE, 2007.

[5] S. Kavalanekar, B. Worthington, Q. Zhang, and S. Vishal. Characterization of storage workload traces from production windows servers. In *IISWC '08*.

[6] Y. Kim, R. Gunasekaran, G. M. Shipman, D. Dillow, Z. Zhang, B. W. Settlemyer, et al. Workload characterization of a leadership class storage cluster. In *Petascale Data Storage Workshop (PDSW), 2010 5th*, pages 1–5. IEEE, 2010.

[7] 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 *Proceedings of the 6th International Workshop on Challenges of Large Applications in Distributed Environments*, pages 15–24. ACM, 2008.

[8] R. Miller, J. Hill, D. A. Dillow, R. Gunasekaran, G. M. . Shipman, and D. Maxwell. Monitoring tools for large scale systems. In *Proceedings of Cray User Group Conference (CUG 2010)*, May 2010.

[9] S. Oral, D. A. Dillow, D. Fuller, J. Hill, D. Leverman, S. S. Vazhkudai, F. Wang, Y. Kim, J. Rogers, J. Simmons, et al. OLCF's 1 TB/s, next-generation lustre file system. In *Proceedings of Cray User Group Conference (CUG 2013)*, 2013.

[10] S. Oral, J. Simmons, J. Hill, D. Leverman, F. Wang, M. Ezell, R. Miller, D. Fuller, R. Gunasekaran, Y. Kim, et al. Best practices and lessons learned from deploying and operating large-scale data-centric parallel file systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 217–228. IEEE Press, 2014.

[11] RedHat Inc. Red Hat Enterprise Linux 6, DM Multipath Configuration and Administration. https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/DM_Multipath/.

[12] A. Riska and E. Riedel. Evaluation of disk-level workloads at different time scales. *SIGMETRICS Perform. Eval. Rev.*, 37(2), 2009.

[13] H. Shan, K. Antypas, and J. Shalf. Characterizing and predicting the i/o performance of hpc applications using a parameterized synthetic benchmark. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, page 42. IEEE Press, 2008.

[14] G. Shipman, D. Dillow, S. Oral, and F. Wang. The spider center wide file system: From concept to reality. In *Proceedings, Cray User Group (CUG) Conference, Atlanta, GA*, 2009.

[15] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar. Synthesizing representative i/o workloads for tpc-h. In *HPCA '04*.