PASSI: A Parallel, Reliable and Scalable Storage Software Infrastructure for Active Storage System and I/O Environments

(Position paper)

Hsing-bung Chen
High Performance Computing Division
Los Alamos National Laboratory
Los Alamos, New Mexico 87545, USA
hbchen@lanl.gov

Song Fu
Department of Computer Science and Engineering
University of North Texas
Denton, Texas 76203, USA
song.fu@unt.edu

***Abstract*—Storage systems are a foundational component of computational, experimental, and observational science today. The ever-growing size of computation and simulation results demands huge storage capacity, which challenges the storage scalability and causes data corruption and disk failure to be commonplace in exascale storage environments. Moreover, the increasing complexity of storage hierarchy and passive storage devices make today's storage systems inefficient, which force the adoption of new storage technologies. In this position paper, we propose a Parallel, Reliable and Scalable Storage Software Infrastructure (PASSI) to support the design and prototyping of next-generation active storage environment. The goal is to meet the scaling and resilience need of extreme scale science by ensuring that storage systems are pervasively intelligent, always available, never lose or damage data and energy-efficient.**

Keywords*—*Ethernet drives; parallel erasure coding; disk failure prediction; metadata management; energy efficiency.

I. Introduction

Computation and simulation advance knowledge in science, energy, and national security. As these simulations have become more accurate and thus realistic, their demands for computational power have also increased. This, in turn, has caused the simulation results to grow in size, increasing the demands for huge storage systems. High performance I/O becomes increasingly critical, as storing and retrieving a large amount of data can greatly affect the overall performance of the system and applications.

The existing storage systems are mostly passive, that is disk drives stage and drain memory/burst buffer checkpoints with little intelligence. With the increasing performance and decreasing cost of processors, storage manufactures are adding more system intelligence at I/O peripherals. *Active storage* aims to expose computation elements within the storage infrastructure for general-purpose computation on data. Although this concept has been appealing for several years and incorporated in the T10 Object-based Storage Device (OSD) standard, a limited number of storage products are available for HPC systems. Moreover, their processing capability is provided at the storage enclosure level, while the majority disk drives are still passive.

Moreover, with exabytes of data to be stored on hundreds of thousands of disk drives, storage scalability becomes a big challenge. More importantly, at such a scale, data corruption and disk failures will be the norm in Exascale storage environments. As data generated by HPC applications, such as checkpoint/restart data sets, get bigger and so do disk drives, the time to rebuild a failed drive is extended, causing tens of hours of data disruption. For the new helium-filled hard drives, due to the larger capacity and the slower increase in performance, the rebuild time of such a disk would be extensive (almost 3 days). In large arrays, a second disk drive can fail before the first is rebuilt, resulting in data loss and significant performance degradation. Dealing with data corruption and disk failures will become the standard mode of operation for storage systems and it should be performed by machines with little human intervention due to the overwhelming storage scale. Moreover, it should cause little data disruption or performance degradation to systems and applications.

In this position paper, we propose a **Parallel, Reliable and Scalable Storage Software Infrastructure (PASSI)** to support the design and prototyping of next-generation active storage system and input/output environments. The goal is to meet the scaling and resilience need of extreme scale science by ensuring that storage systems are *intelligent* (at disk drive level), *always available* and *never lose or damage data*.

PASSI (in Fig. 1) explores the new-generation *Ethernet drives* to provide smart, low-power and scaling storage structure, integrates parallel erasure coding for high-performance data integrity, disk failure prediction for proactive data rescue, and object storage with balanced data placement, and supports metadata technologies for high performance file systems. *These storage services are offloaded to the smart Ethernet drives*, which makes PASSI unique, novel and attractive.

The rest of this paper is organized as follows. We present Ethernet connected drives in Section II. In Section III, we present the design of Parallel Erasure Coding and Parallel Data Placement software systems. In Section IV, we propose an S^3 (Self-Monitoring, Self-Managing and Self-Healing) software system in PASSI to address the disk failure challenge and

978-1-4673-8590-9/15/$31.00 ©2015 IEEE

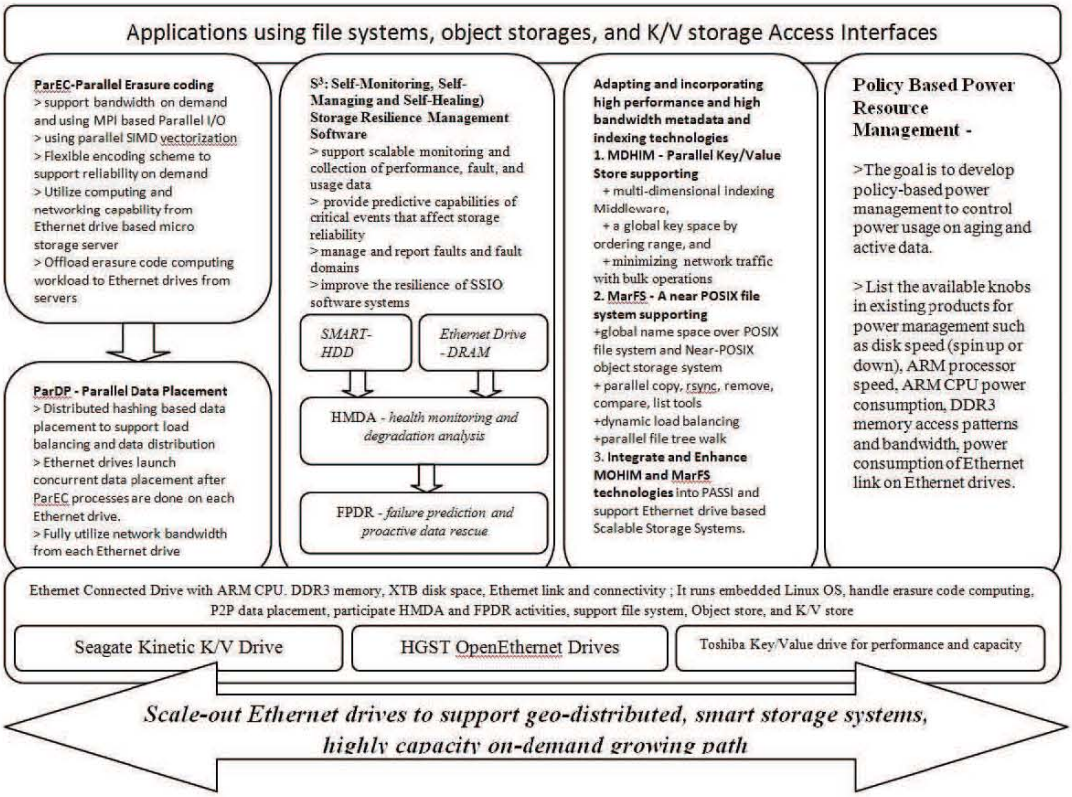

Fig. 1: PASSI software infrastructure.

TABLE 1: Technical features of three different Ethernet connected products.

	ARM	Memory	OS/Software	Disk/HDD	Ethernet	Platform	SSD
Seagate	32-bit, 700MHZ, 1-core	512MB	K/V store	4TB	One active and one standby	Close platform, API or using LLVM	Planning
HGST	32-bit, 1GHZ, 1-core	2GB DD3	Embedded Linux OS	4TB/8TB	One active	Open platform, run applications	Planning
Toshiba	64-bit Quad-core	N/A	K/V store Linux OS	4TB: two 2.5" HDD for performance	Two Ethernet links	Same as Kinetic drive	2 SSDs for performance
				One 3.5" SMR drive for capacity			N/A

DRAM errors of Ethernet drives. In Section V, we describe how we plan to leverage LANL's MDHIM (a Parallel Key/Value store) and MarFS (a near-POSIX metadata technology for Exascale storage systems) in PASSI. In Section VI, we present a policy based power management approach for archive storage systems. We conclude this paper and outline our current and future development activities in Section VII.

II. Build Active and Scaling Storage Systems Using Smart Ethernet-Connected Drives

Ethernet Drive technology provides a new paradigm shift of storage architecture. Ethernet Drives are storage devices that can support software-defined storage (SDS). There are three different Ethernet connected disk drive technologies (Table 1). These products possess attractive features: 1) An Ethernet drive has embedded low-power ARM CPU and memory. 2) It runs Linux OS on the drive. 3) It can run data processing services closer to the storage media. 4) It has Ethernet access link and IP based connectivity. 5) It has a high capacity disk space and/or SSD storage. Ethernet drives enable us to eliminate high-end storage server requirements, cut down power consumption of data storage systems, and move application services closer to storage media.

The new Ethernet drives do not merely use Ethernet as a new connection interface; they move the communication

protocols from commands on read-and-write data-blocks to a higher level of abstraction. Ethernet drives can offload traditional logical block address (LBA) management and provide new data placement schemes for dispersing object data. This new technology is intended to simplify storage system software stack, allow drive-to-drive data movement, and reduce significant drive I/O activities from servers. In this paper, we use Ethernet drives to build a scale-out storage system and design PASSI to achieve self-management, self-monitoring, and self-healing on this active storage system.

A. Seagate's Kinetic Drives

Seagate's Kinetic drive [1, 11] employs a new access interface and APIs. It uses a key-value data approach to store data at the object level on the drives. Kinetic drive's access interface is an attempt to bypass many layers of the current OS file stack and provide an efficient software stack to accommodate current and future application demands. Seagate also provides an LLVM access interface [27], allowing us to run non-key/value storage services on Kinetic drives.

B. HGST's OpenEthernet Drives

HGST's Open Ethernet drive [2] (aka OED) has been designed as a micro-storage server. An OED has one single-core 32-bit 1 GHZ ARM processor, 2 GB DDR3 DRAM memory, one single Gigabit Ethernet link, and 4 TB disk space. It runs an embedded Linux OS and has extra DRAM memory space and fast processor to run applications. Moreover, HGST's OED architecture enables us to take advantage of Ethernet drives without having to implement the proposed PASSI software following a vendor-specific API and architecture.

C. Toshiba's Key/Value Drives

Toshiba has released two Ethernet connected drive products [3, 4, 5]. The new performance based Ethernet drive technology combines two 2.5-inches high-capacity SATA disk drives, two high-performance M.2 SSDs, two Gigabit Ethernet ports, a four-core 64-bit ARM processor, and the Linux OS in a 3.5-inch hard drive enclosure. The other capacity based Ethernet drive uses an SMR disk for capacity-centric purpose. Toshiba's Ethernet Drives adapt Kinetic Drive's interface assess APIs, which helps us to port our PASSI software between the two products.

D. Ethernet Drives Meet Scale-Out Requirements

Big data applications rely on being able to easily and inexpensively create, manage, administrate, and migrate unstructured data. Legacy file and storage architectures cannot meet with these requirements, and thus cannot provide a high performance and cost-effective solution to Exascale computing [10]. The new Ethernet drive technologies provide a new, promising direction to address the extreme scale-out challenge for bulk storage applications. These include applications of primary storage, unstructured data, information governance, analytical data, active archival, and cold storage.

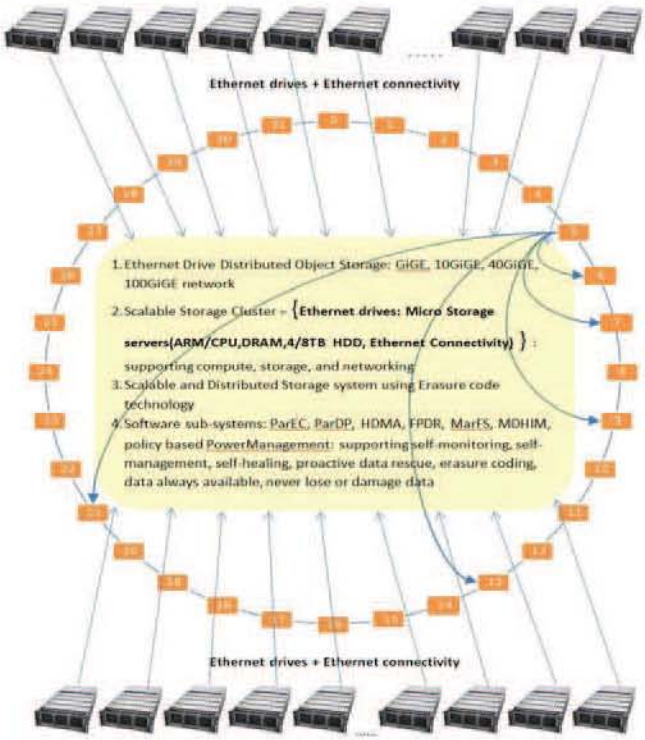

Fig. 2: Architecture of smart Ethernet-Drive storage system.

III. Design Parallel Erasure Coding and Balance Data Placement for PASSI

Erasure-coded storage systems become increasingly popular for HPC due to their cost-effectiveness in storage space saving and high fault resilience capability [28]. However, the existing systems use a single machine or a multi-threaded approach to perform erasure coding, which is not scalable to process the gigantic checkpoint/restart data sets for Exascale science applications. To address this scaling challenge, we propose a *Parallel Erasure Coding (ParEC)* technique which can encode and decode small chunks of a big data object in parallel on Ethernet drives, and a *Parallel Data Placement (ParDP)* technique that distributes erasure segments evenly to all available Ethernet drives in a system.

A. ParEC - Parallel Erasure Coding

Multi-core processor design has been implemented in today's CPU technology. Each multi-core processor has its own embedded SIMD subsystems [16]. It is a challenging task to achieve a high parallelism for erasure coding and maximize CPU utilization. We notice that the portion of System-IDLE state dominates the overall processing time and energy cost compared with I/O and erasure coding processes. When encoding and decoding a very big file or dataset, using a single-process approach is not capable of achieving sustained bandwidth for erasure code computation and data movement in an HPC campaign storage system. We need to fully utilize the processing capacity of a storage cluster and provide parallel I/O capability. To this end, we propose a parallel erasure code

technology and apply it to very large files and dataset, such as checkpoint-restart data and simulation results, for high performance scientific simulation.

We extend the existing single-process or single-thread erasure coding method by adding task/function parallelism and data parallelism. We propose a hybrid parallel approach, named Parallel Erasure Coding (ParEC). It uses MPI parallel I/O for task parallelism and processor's SIMD extensions for data parallelism [28, 29]. We apply this hybrid parallelism to perform compute-intensive erasure coding and to provide a high performance, energy-saving, and cost-effective solution for future all-disk based HPC campaign storage systems. We demonstrate that ParEC can significantly reduce the power consumption, improve coding bandwidth, and speed up data placement. We evaluate ParEC on Ethernet drives equipped with 32-bit ARM CPUs and conduct performance and feasibility studies. We compare the vectorization technologies of Intel X86/SIMD and ARM/Neon in terms of performance and energy use for erasure code based HPC campaign storage systems.

With the combination of data parallelism based on SIMD computation and task parallelism based on MPI parallel I/O, parallel erasure coding software can improve the performance and scalability of erasure coding techniques, reduce the archival overhead of extreme-scale scientific data, and reduce power and energy consumption. with ParEC, we efficiently run more scientific simulations on extreme-scale computers and thus enhance the system utilization. Moreover, the saved power and energy can be exploited by other components of the system, which further improves the system throughput in a cost-effective manner.

B. ParDP - Parallel Data Placement

For a file with K data segments, ParEC produces M redundant code segments. The $K+M$ encoding scheme can tolerate a loss of up to M data segments and/or code segments. These segments are placed on $K+M$ Ethernet drives according to a placement policy. The reconstruction strategy in case of disk failures is also addressed in the placement policy.

We leverage the Ethernet drive technology and Peer-to-Peer data placement protocols to build a distributed data storage environment for PASSI. We propose a parallel data placement mechanism called ParDP. ParDP is responsible for placing the data and code segments generated from ParEC onto Ethernet drives. More specifically, we explore distributed hashing methods, in which each Ethernet drive is responsible for a data region on a circular space, called ring as shown in Fig. 2. To address the limitation of traditional consistent hashing methods for Exascale storage systems, we enhance the consistent hashing by assigning multiple points in the ring to an Ethernet drive and using the concept of virtual drives. A virtual drive is like a physical drive, but each physical drive can be responsible for multiple virtual drives, depending on its capacity. This addresses the heterogeneity issue. When an Ethernet drive is predicted to fail (see Section IV), the data stored on that drive are dispersed to other available drives proactively and smoothly without data and computation disruption, probably at different geo-locations. Data blocks can

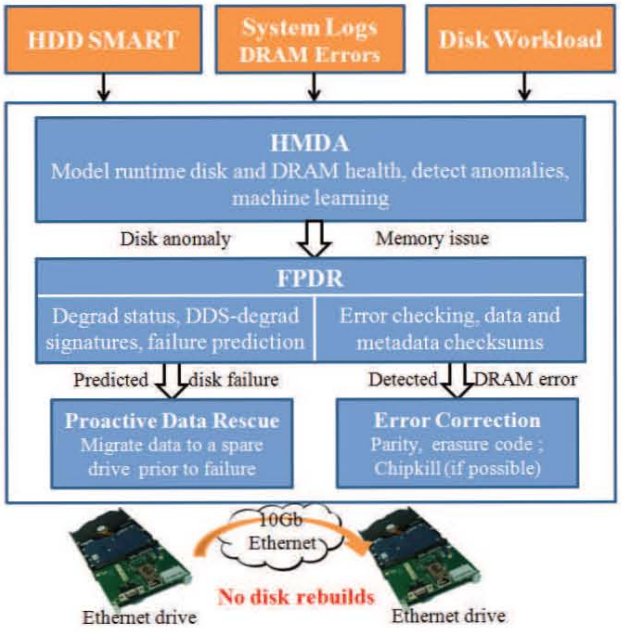

Fig. 3: Major components of S^3.

be recovered through parallel rebuilds on Ethernet drives using ParEC to remedy data corruptions. ParDP also records metadata, index, and data placement information for MarFS and MDHIM to manage these metadata (see Section V).

IV. S^3 (Self-Monitoring, Self-Managing and Self-Healing) Storage Resilience Management

The design of S^3 storage resilience management software must satisfy a number of requirements: 1) They should support scalable monitoring and collection of performance, fault, and usage data. Thus online analytics can provide key data to storage managers with low overhead. 2) They should provide predictive capabilities of critical events that affect storage reliability. The collected parameters should be correlated with Ethernet drive characteristics to gain a correct assessment of the drive state. 3) They should efficiently manage and report faults and fault domains. Thus proactive data rescue mechanisms can be employed promptly to heal the faults. 4) They should improve the resilience of storage software systems. The vulnerability of storage software to soft errors should be characterized and mitigated.

We propose a set of novel techniques for storage resilience assurance that integrate disk health monitoring, disk degradation detection, disk failure prediction, proactive data rescue and memory error detection and correction with distributed controls over Ethernet drives. Machine learning technologies are explored extensively to automate these management tasks. The novelty of S^3 lies in that it is the first of its kind that can automatically discover different *types* of disk failures and accurately forecast the future occurrences of failures for *each type*. This is based on a novel concept that we propose, called *disk degradation signatures* [8], which characterize and quantify the deterioration process of drives

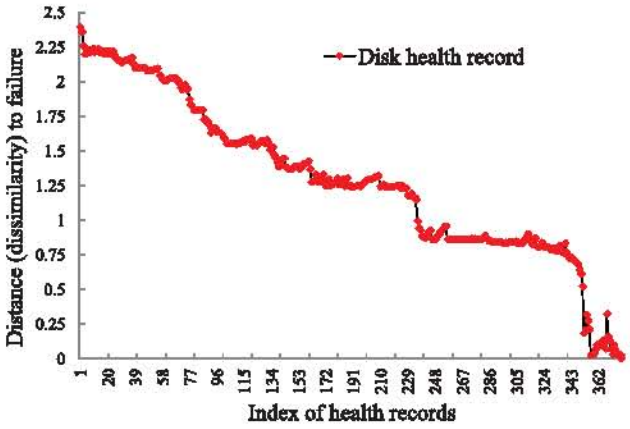

Fig. 4: Disk performance degradation of a disk failure.

from having disk errors to becoming failed. We explore the embedded processors and memory on Ethernet drives to perform in situ analysis of the real-time disk health data, and the on-drive Ethernet ports to exchange analysis results for higher-level aggregation and analysis. We also investigate the correlation between disk workloads and disk degradation, which guides us to develop reliability-aware workload scheduling strategies [17, 32].

The embedded DRAM on Ethernet drives may experience errors and faults at runtime [24]. We propose software based memory integrity checking and correction. We check data integrity on each memory access by using error-checking code. We use internal data and metadata checksums to detect silent data corruptions. In addition, the PASSI software is vulnerable to soft errors. To ensure the resilience of PASSI software system, we leverage soft error injection tools, such as our F-SEFI [30, 31], to evaluate the vulnerability of PASSI to SDCs and design effective mechanisms to mitigate these vulnerabilities.

We propose two sub-systems of S^3: *health monitoring and degradation analysis (HMDA)* and *failure prediction and proactive data rescue (FPDR)*, as shown in Fig. 3.

A. HMDA- Health Monitoring and Degradation Analysis Sub-system

An HMDA daemon on each drive retrieves disk SMART readings, memory and system logs periodically from the attached disk drive. These runtime data are machine-learned to build disk and memory health models and detect anomalous behaviors [8][18][19][20][21][22][23]. Aggregated disk health data are collected through Ethernet connections from these active drives to HMDA backend processes for higher-level analysis, where disk replacement and memory fault events are automatically cross-referenced with the detected anomalies and monitored health data to build and update *disk degradation signature* and *memory error* models [8][24]. Fig. 4 shows the performance degradation of a disk drive prior to a failure.

B. FPDR - Failure Prediction and Proactive data Rescue Sub-system

FPDR leverages regression learning and ensemble learning to track disks' degradation and memory health status, and apply the degradation signatures to forecast when disk failures will occur and detect memory errors. FPDR finds a spare drive and performs *proactive data rescue* for a failing drive, from which the data are migrated to the spare drive in background prior to a disk failure. FPDR also corrects the detected memory errors. *Ultimately, we apply FPDR to reduce and even avoid disk-rebuild overhead commonly seen in today's storage systems and ensure storage systems are always available.*

C. Research Issues of HMDA and FPDR Sub-systems

There are critical issues that we investigate for efficient, scalable storage resilience management. 1) How to store the collected runtime disk and memory health data? The runtime disk SMART records and system logs are saved on the attached disk of each Ethernet drive. To limit the footprint of these data, we employ a *monitoring window*, whose size is configurable. Only the health data inside the current monitoring window are stored locally. Those that are prior to the current window have been learned and incorporated in the disk degradation and memory error models. Thus the raw data can be discarded except for those that are correlated with critical disk and memory errors, which are transferred to the HMDA backend processes for bookkeeping and aggregation. We evaluate different sizes of the monitoring windows and exploiting machine learning to select the best one at runtime. 2) How to handle false negative disk failure predictions? Our disk failure predictors keep tracking the degradation status of each drive. Since disks follow gradual degradations in a production environment, our preliminary results show the false negative rate is very low. Some of the false negatives can be logical failures, e.g., caused by corrupted files or file system faults. In these cases, the drives have no physical damage. Diagnostic tools are used to find the software problems and correct them if possible. For missed disk hardware failures, disk rebuild or parallel disk rebuild is performed. An alternative strategy is to explore parallel erasure coding (ParEC) to rebuild data blocks. We expect the performance degradation to be little because those failures are very rare. 3) How to detect memory errors and silent data corruption in Ethernet driver? Non-chipkill memory technology is currently used in Ethernet drives. Without chipkill and an end-to-end data protection technology, data corruption can go unnoticed and recovery is difficult and costly, or even impossible to perform. Without end-to-end integrity checking, these silent data corruptions can lead to unexpected and unexplained problems [25]. We apply software based memory integrity checking and correction [26] in addition to DRAM ECC. We check data integrity during memory accesses by using error-checking code. We use internal data and metadata checksums to detect silent data corruptions.

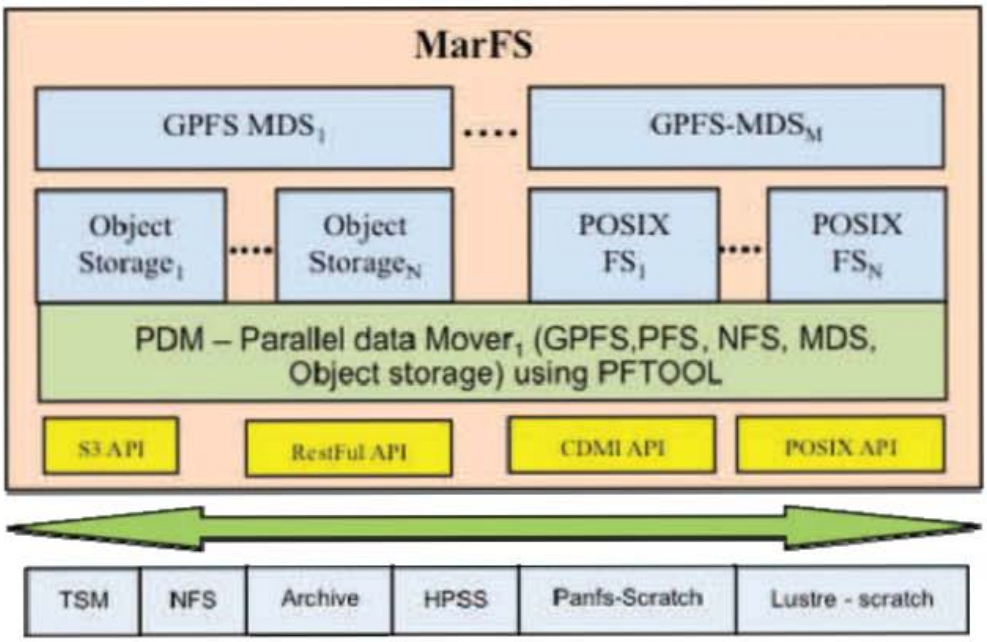

Fig. 5: MarFS system architecture with metadata management.

V. Metadata Support in PASSI

MarFS [6] and MDHIM [7] are two key technologies invented and designed by LANL's researchers to support metadata for HPC storage systems.

A. Integration with MarFS – A Hybrid Metadata Service Framework for Non_POSIX and POSIX Storage Systems

Metadata is commonly divided into two broad categories: *structural metadata*, which concerns the design and specification of data structures, and *descriptive metadata*, which comprises all other associated information such as creator, meaning, intended uses, provenance, associations, and context [10]. MarFS is designed to integrate both structural metadata and descriptive metadata [6].

MarFS is a near-POSIX file system using scale-out commercial storage systems as data stores and POSIX file systems for global namespace-based metadata services. Fig. 5 presents its architecture. MarFS provides the scalability of a global namespace by sewing multiple POSIX file systems together as parts of hierarchical file trees from POSIX systems and as parts of multiple single dimensional containers or buckets used in object-based storage systems. A distinguishing feature of MarFS is that it separates metadata information from the data store. This feature creates a fully portable and dispersible operation environment for both metadata and data. Metadata are moved and duplicated on multiple geo-locations so that the system can maintain a highly resilient backup for metadata information. We use erasure code-based object storage system as the data store. Extended file attributes (xattr) are used in MarFS to record the embedded information of the data store, such as data location. MarFS enables the PASSI middleware to provide metadata services for both non-POSIX and POSIX HPC storage.

We integrate the MarFS framework into PASSI software infrastructure and use MarFS technology to support high performance metadata services and manage erasure code based Ethernet-drive storage systems.

B. Integration with MHDIM – A Parallel Key/Value Framework for HPC Storage

We have developed a Multi-Dimensional Hashing Indexing Middleware (MDHIM) [7], a parallel key-value store framework designed for HPC systems. MDHIM differs from other key-value stores in that it supports HPC networks natively, uses dynamic servers that start and end with applications, works with pluggable database backends, orders a global key space by range instead of hash, uses cursor type operations, exposes multiple dimensions, and minimizes network traffic with bulk operations. MDHIM is designed specifically for HPC supercomputers. It works with Infiniband and the variants, takes advantage of the massive computational resources on supercomputers, and runs as a library in HPC applications.

MDHIM provides the capability for us to define how to distribute data across the range servers with a configurable range size. The entire key space is a series of ranges that are addressable between 1 to 232-1 and each range can contain at most 264 keys. The size of the range determines which range servers are responsible for which keys. A partitioner decides which range server handles a key by running the mapping

algorithms implemented for the different key types. These mappings may not be optimal for every workload, however; they provide a reasonable default mapping.

Multi-dimensionality has been implemented, which allows us to create any number of secondary index instances. Each index instance creates a new data store instance, if it does not exist, separates the indexes and allows them to operate independently. Data stores can be implemented to create an index local to the database instead of a new database. Secondary keys can be inserted at the same time as the primary key, or anytime after the primary key has been inserted.

An MDHIM prototype has been written in MPI to take advantage of high-speed interconnects natively and to be easily embedded into an HPC application. *It is highly scalable and it supports multiple dimensions, pluggable data stores, cursor type operations, and bulk messages, which facilitates PASSI's interaction with key-value stores.* We integrate the MDHIM framework into PASSI software infrastructure and use MDHIM framework to build a multi-dimensional index system to manage and maintain information of ParEC's erasure coding and ParDP's data placement.

VI. Policy based Power management

The goal is to develop policy-based power management strategies to control power usage on aging and active data. The available knobs in existing products for power management include disk speed (spin up or down), ARM processor speed, ARM CPU power consumption, DDR3 memory access patterns and bandwidth, and power consumption of Ethernet link on Ethernet drives. The mechanisms may include the selection from multiple predefined firmware-level configurations.

A. Design Requirements

The management of massive smart Ethernet-connected drives is challenging in the following aspects. The simultaneous spinning-up or -down of drives can cause power spikes that increases the electricity bill and affects reliability. Most disk power management systems today are based on spinning up/down 10 disks at a time. In a massive scale environment we might want to power up and down 1,000 disks at a time. This is something a bit different to be managed than what today's solutions can provide [12][13][14][15].

B. Proposed Approaches

A policy can be described in two different ways. One way is to define a policy from its operational view. For example, spinning down the disk to 5,400 rpm when the disk utilization is below 70%. Another way is to define a policy from its denotational view. For example, maintaining the disk performance to be 100MB/s or above. The denotational-based policy only specifies the end goal, but lets the systems software determine how to achieve the goal. Both views can be applied to the case of power limiting as well. In this research, we study both types of policy, starting with policy based on the operational view. For policy based on the denotational view, a model-driven approach is taken in the study.

VII. Conclusions and Future works

In this position paper, we present PASSI, a parallel, reliable and scalable storage software infrastructure, exploring the smart Ethernet drives. This research integrates storage system analysis, algorithm design with system implementation. The develop PASSI mechanisms and tools will be tested and deployed on ORNL and LANL production systems, if successful. This project produces enabling technologies to build high-performance, scalable and energy-efficient storage systems, where the storage services can be performed by smart disk drives in a distributed, cost-effective and reliable fashion. This makes a significant impact to scientific computing at the extreme scale by ensuring the storage systems can be counted on to always be available and never lose or damage data, allowing large scientific applications to run to a correct solution in a timely and efficient manner.

The developed PASSI software can also provide insights into the runtime power and energy use of large-scale HPC storage systems, which facilitates intelligent power management and energy saving techniques to address the power challenge of Exascale computers.

Acknowledgment

We would like to thank the anonymous reviewers for their constructive comments and suggestions.

This work was performed at the Ultrascale Systems Research Center (USRC) at Los Alamos National Laboratory, supported by the U.S. Department of Energy contract AC52-06NA25396. This work was supported in part by the US Department of Energy and LANL ASCI funding. We thank the technical support from LANL's HPC-5 group and HPC-3 group. The publication has been assigned the LANL identifier LA-UR-15-25086.

References

[1] Seagate's Kinetic Drive - http://seagate.com

[2] HGST's OpenEthernet Drive - http://www.hgst.com

[3] Toshiba's Key/Value Drive - http://toshiba.com

[4] Toshiba Key-Value Drive White Paper, May2015

[5] Toshiba - Introduction to Toshiba Key Value Drive Technology, May 2015

[6] Gary Grider, MarsFS: A near-POSIX Scalable File System using Distributed Object Stores , 2015 IEEE International Conference on Massive Storage Systems and Technology (MSST 2015) Conference

[7] Hugh Greenburg, MDHIM: A Parallel Key/Value Framework for HPC, USENIX HotStorage Workshops, 2015.

[8] S. Huang, S. Fu, Q. Zhang and W. Shi. "Characterizing Disk Failures with Quantified Disk Degradation Signatures: An Early Experience". In *Proceedings of IEEE International Symposium on Workload Characterization (IISWC)*, 2015.

[9] Hsing-bung Chen, An Empricial Study of Performance, Power Consumption, and Energy Cost of Erasure Code Computing for HPC storage systems , 2015 IEEE International Conference on Networking, Architecture, and Storages (NAS 15)

[10] Storage Systems and Input/Output to Support Extreme Scale Science, Report of the DOE Workshops on Storage Systems and Input/Output, Dec. 8-11, 2014

[11] James Hughes, Seagate Kinetic Open Storage Platform, 2014 IEEE International Conference on Massive Storage Systems and Technology (MSST 2014) Conference

[12] Chung-Hsing Hsu and Stephen W. Poole, Power Measurement for High Performance Computing: State of the Art, In 2011 International Conference Green Computing Conference and Workshops (IGC)

[13] James William Smith, Ali Khajeh-Hosseini, Jonathan Stuart Ward, and Ian Sommerville, CloudMonitor: Profiling Power Usage, In 2012 IEEE 5th International Conference on Cloud Computing

[14] Jack Dongarra, Hatem Ltaief, Piotr Luszczek, and Vincent M. Weaver, Energy Footprint of Advanced Dense Numerical Linear Algebra Using Tile Algorithms on Multicore Architectures, In 2012 Second International Conference on Cloud and Green Computing (CGC)

[15] Chung-Hsing Hsu, Jacob Combs, Jolie Nazor, Fabian Santiago, Rachelle Thysell, Suzanne Rivoire. Application Power Signature Analysis, HPPAC 2014 - The 10th Workshop on High-Performance, Power-Aware Computing

[16]Aleksei Marov and Sergei Platonov, Effective method for coding and decoding RS codes using SIMD instructions, 2014 High performance Extreme Computing Conference

[17]Philip Carns, Robert Latham, Robert Ross, Kamil Iskra, Samuel Lang, and Katherine Riley, 24/7 Characterization of Petascale I/O Workloads, 2014 Parallel Processing: 20th International Conference Euro-Par

[18] G. Hughes, J. Murray, K. Kruetz-Delgado, and C. Elkan. "Improved disk drive failure warnings". *IEEE Transactions on Reliability*, 51(3):350–357, 2002.

[19] J. Li, X. Ji, Y. Jia, B. Zhu, G. Wang, Z. Li, and X. Liu. "Hard drive failure prediction using classification and regression trees". In *Proceedings of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2014.

[20] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado. "Hard drive failure prediction using non-parametric statistical methods". In *Proceedings of International Conference on Artificial Neural Networks*, 2003.

[21] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado. "Machine learning methods for predicting failures in hard drives: A multiple-instance application". *Journal of Machine Learning Research*, 6:783–816, 2005.

[22] B. Eckart, X. Chen, X. He, and S. L. Scott. "Failure prediction models for proactive fault tolerance within storage systems". In *Proceedings of IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems (MASCOTS)*, 2008.

[23] J. Gray and C. V. Ingen. "Empirical measurements of disk failure rates and error rates". Technical Report MSR-TR-2005-166, *Microsoft Research*, 2005.

[24] V. Sridharan, N. DeBardeleben, S. Blanchard, K. B. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi. "Memory Errors in Modern Systems: The Good, The Bad, and The Ugly". In *Proceedings of ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2015.

[25] Preventing Silent Data Corruption - Using Emulex Host Bus Adapters, EMC VMAX and Oracle Linux , Oracle White paper, Sept. 2013

[26] Yupu Zhang, Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau , End-to-end Data Integrity for File Systems: A ZFS Case Study, In *Proceedings of USENIX International Conference on File and Storage Technologies (FAST)*, 2010

[27] Chris Lattner Vikram Adve, LLVM: A Compilation Framework for Lifelong Program Analysis & Transformation, 2004 Proceedings of the international symposium on Code generation and optimization.

[28] James S. Plank, Erasure Codes for Storage Systems: A Brief Primer ; login: the Usenix Magazine, December 2013, Volume 38, Number 6. Usenix Association.

[29] James S. Plank, Kevin M. Greenan and Ethan L. Miller. Screaming Fast Galois Field Arithmetic Using Intel SIMD Instructions, FAST 2013: 11th USENIX Conference on File and Storage Technologies.

[30] Q. Guan, S. Fu, N. DeBardeleben and S. Blanchard. F-SEFI: A Fine-grained Soft Error Fault Injection Tool for Profiling Application Vulnerability, In *Proceedings of 28th IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, 2014.

[31] Q. Guan, N. DeBardeleben, S. Blanchard and S. Fu. Addressing Statistical Significance of Fault Injection: Empirical Studies of the Soft Error Susceptibility, In *the 21st IEEE/IFIP International Symposium on Dependable Computing (PRDC)*, 2015.

[32] S. Huang, S. Fu, N. DeBardeleben, Q. Guan and C. Xu. Differentiated Failure Remediation with Action Selection for Resilient Computing, In *the 21st IEEE/IFIP International Symposium on Dependable Computing (PRDC)*, 2015.