Scaling High-Performance Parallel File Systems in the Cloud

Jimmy Beckett[†] Eric Kim[†] Evan Stanton[†] David Liu[†] Vishank Rughwani[†] Min Hwang[†]
Joaquin Chung[‡] Rob Fatland[†§] Nam Pho[†§]
[†]Research Computing Group and [§]eScience Institute, University of Washington, Seattle, WA
[‡]Data Science and Learning Division, Argonne National Laboratory, Lemont, IL

***Abstract*—On the path to exascale computing, high-performance parallel file systems are an integral infrastructure that allow research workflows to scale. Among supercomputing sites across the globe these include Lustre, BeeGFS, and Gluster. The key to performance, other than the various nuances of these file systems, is how the underlying servers are architected and configured. This project aims to develop infrastructure as code to scalably deploy and tune each parallel file system in an array of environments, specifically public and academic clouds to explore the feasibility of hybrid or pure cloud solutions for high-performance workflows. This work will qualify these systems with an array of storage benchmarks including Bonnie, IOzone, IOR, and IO-500. We will also utilize scientific workflows representative of various research domains including precision medicine and machine learning.**

I. Challenges

A. Performance

While the concept of distributing data reads and writes across multiple servers to scale storage performance and bandwidth is intuitive, various file system designs can be leveraged to different degrees through their unique architectures. Lustre and BeeGFS scale by separating metadata and data operations onto distinct servers, while distributed file systems like Gluster make use of identical nodes where metadata and data operations are not separated. These distinct architectures necessitate design considerations for achieving optimal performance. The deliverable of this work is a data driven, systematic exploration of those considerations and their performance outcomes or trade offs.

B. Reproducibility

Parallel file systems are agnostic to the underlying resource, whether bare-metal or virtualized, on-premise or the cloud. Through the course of evaluating design considerations (e.g., ratio of metadata to data servers, CPU cores per server, file striping setting, individual OS kernel parameters) we will create and optimize an infrastructure as code utility to automate and configure the deployment of storage systems targeted by this study. Large IT deployment tools already exist for such use, such as Ansible or Puppet for configuration management of systems at scale. Deployment code created in the course of this experimentation will be made available to the scientific computing community.

II. Approach

A. Parallel File Systems

Lustre, BeeGFS, and Gluster are scalable storage solutions with rich development histories out of major supercomputing centers. In the case of Lustre and BeeGFS, they are the top ranked open-source file system deployments on the June 2018 IO-500 list. Gluster represents a different scale out approach using distributed hash tables worth further study.

Instance selection for storage sub-system components is critical towards overall performance. Our public (e.g., AWS, Azure, GCP) and academic (e.g., Jetstream, Chameleon) clouds provide dozens of options with different underlying hardware implementations, all along the critical path to final performance. Our goal is to explore combinations of instances within storage systems built in the cloud as well as the OS and file system configurations at the software layer. The control system is a 1 Petabyte, two NSD node GPFS appliance using default configurations.

B. Benchmarks

Benchmarking a file system can be considered an art with multiple tools and various standards to choose from. While not consistently correlated with real-world application performance, accepted file system metrics are input/output operations per second (IOPS) for random and sequential access of various file sizes as well as total bandwidth of data transfers. In our study we intend to use these metrics as measured by established applications such as Bonnie, IOzone, or IOR and modern approaches in IO-500 [1]. We also consider real-world scientific workflows such as genomic variant calling in precision medicine (e.g., GATK [2]) and image recognition in machine learning (e.g., ResNet-50 [3]) as defined by total runtime for identical workflows while varying the underlying storage file system and configuration.

In initial testing of our control system using GPFS we averaged 111,320 $\pm$ 33,736 IOPS over 1,000 iterations simulating a 512 kB file in 4 kB segments. Maximum sequential read and write bandwidth for these same tests reported 30.2 $\pm$ 7.1 Gbps and 17.5 $\pm$ 3.8 Gbps, respectively. Testing was done while the storage system supported normal, production workloads and therefore represents a real world average of excess performance capacity and basis for further analysis.

References

[1] Julian Kunkel, John Bent, Jay Lofstead, George S. Markomanolis. 2017. "Establishing the IO-500 Benchmark." In *2nd J. Intl. Workshop on Parallel Data Storage & Data Intensive Scalable Computing Systems*.

[2] McKenna, Aaron, et al. 2010. "The Genome Analysis Toolkit: A MapReduce Framework for Analyzing next-Generation DNA Sequencing Data." *Genome Research* 20 (9): 12971303.

[3] He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep Residual Learning for Image Recognition." In 2016 *IEEE Conf. on Comp. Vis. and Pattern Recog. (CVPR)*, 77078.