# Enabling Scientific Application I/O on Cloud FileSystems

Milo Polte (student)[1], Esteban Molina-Estolano (student)[2], John Bent[3], Scott Brandt[2], Garth Gibson[1], Maya Gokhale[4], Carlos Maltzahn[2], and Meghan Wingate[3]

[1]Carnegie Mellon University
[2]UC Santa Cruz
[3]Los Alamos National Laboratory
[4]Lawrence Livermore National Laboratory

Today, the storage of many of the world's largest computing clusters is being supported not through traditional parallel filesystems, such as Lustre[7], the Parallel Virtual Filesystem (PVFS)[4], or PanFS[6], but through "cloud" filesystems such as GoogleFS[5], the Hadoop Distributed Filesystem (HDFS)[2], and the Amazon Simple Storage Service (Amazon S3).[1] Although these filesystems provide appropriate performance and resilience for web apps, their APIs are often unsuitable for the I/O patterns of a scientific application. Scientific applications typically use a POSIX interface, often with additional support for semantics such as concurrent writers or out of order writes. These applications cannot presently run at all on a cloud filesystem, since these filesystems lack the necessary semantics. We would like to remedy this deficiency: running scientific applications on cloud filesystems would allow sites with investments in cloud filesystems to leverage existing hardware to run scientific computing applications as well.

Previously, the authors have designed a thin interposition layer, the Parallel Log-structured Filesystem (PLFS)[3], to improve the I/O speed of scientific applications running on a parallel filesystem. PLFS transparently decouples inefficient, concurrent writers accessing a shared file into efficient, log-style writers maintaining individual data files, while still providing the application with a view of a single, flat file.

Although this previously work was motivated by performance rather than enabling functionality, we believe the same techniques can be adapted to support concurrent writers in a cloud filesystem. In this work, we will extend PLFS to run on the Hadoop filesystem and add shared write semantics to it. This will enable scientific applications, such as simulation checkpointing, to run unmodified on top of the Hadoop filesystem. We plan to compare the performance against the same scientific apps running on PVFS (a common parallel filesystem used at National Labs) on the same hardware and evaluate the trade-offs in performance.

## References

[1] Amazon. Amazon simple storage service (amazon s3). http://aws.amazon.com/s3/.

[2] Apache. Welcome to hadoop distributed filesystem! http://hadoop.apache.org/hdfs/.

[3] John Bent, Garth Gibson, Gary Grider, Ben McClelland, Paul Nowoczynski, James Nunez, Milo Polte, and Meghan Wingate. Plfs: a checkpoint filesystem for parallel applications. In *SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, pages 1–12, New York, NY, USA, 2009. ACM.

[4] Philip H. Carns, Iii, Robert B. Ross, and Rajeev Thakur. In *ALS'00: Proceedings of the 4th annual Linux Showcase & Conference*, Berkeley, CA, USA.

[5] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. *SIGOPS Oper. Syst. Rev.*, 37(5):29–43, 2003.

[6] David Nagle, Denis Serenyi, and Abbie Matthews. The panasas activescale storage cluster: Delivering scalable high bandwidth storage. In *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 53, Washington, DC, USA, 2004. IEEE Computer Society.

[7] Philip Schwan. Lustre: Building a file system for 1,000-node clusters. In *Proceedings of the 2003 Linux Symposium*, July 2003.