Haceph:
Scalable Metadata Management for Hadoop using Ceph

Esteban Molina-Estolano, Carlos Maltzahn, Ben Reed, and Scott A. Brandt
UC Santa Cruz and Yahoo!, Inc.
{eestolan,carlosm,scott}@cs.ucsc.edu, breed@yahoo-inc.com
No demo.

Hadoop has become a hugely popular platform for large-scale data analysis. This popularity poses ever greater demands on the scalability and functionality of Hadoop, and has revealed an important architectural limitation of its underlying file system: HDFS provides only one *name node* which has to store the entire file system name space in main memory. This limitation puts a hard limit on the amount of metadata, in particular the number of files, HDFS can store. Large clusters frequently run out of capacity at the name node to track new files even though there is plenty of storage capacity at the data nodes. The single name node also creates a single point of failure and a potential performance bottleneck for workloads that require relatively large amounts of metadata manipulations such as opening and closing of files. The single name node limitation is well-recognized in the Hadoop user and developer community (see for example [9]).

One solution is to distribute the functionality of the name node across multiple nodes by statically partition the name space either by subtree partitioning or by hashing of individual directories. A better approach—and the one we pursue here—is to use dynamic subtree partitioning [8] which allows busy nodes to shed popular subtrees to less busy nodes while preserving access locality.

We are working on making Ceph [5] available as an alternative to HDFS with equivalent or better performance. Ceph is an object-based parallel file system with a number of features that make it an ideal storage system candidate for Hadoop: Ceph's *scalable metadata server* [8] can be distributed over hundreds of nodes while providing consistent, reliable, and high-performance metadata service using dynamic subtree partitioning with close to linear scalability. (2) Each file can specify its own *striping strategy* and object size. Flexible striping strategies and object sizes are important tuning parameter for Hadoop workloads [3, 4, 2]. (3) Data is stored on up to 10,000s of nodes which export a single, *reliable object service* [7] with a flat name space of object IDs, not unlike Amazon's Simple Storage Service (S3) [1]. Changes in the storage cluster size cause automatic and fast failure recovery and rebalancing of data with no interruption of service and minimal data movement, making Ceph suitable for very large deployments. (4) The state of the entire storage cluster, including data placement, failed nodes, and recovery state, has a very compact representation due to calculated placement [6] as opposed to allocation tables, and is known in every part of Ceph. As in HDFS, Hadoop's scheduler can take advantage of this information to place mapping close to where the data resides. (5) Ceph is an open source project (ceph.newdream.net) written in C++ that started as a Ph.D. research project at UC Santa Cruz over four years ago and has been under heavy development ever since. A Hadoop module for integrating Ceph into Hadoop is in development since release 0.12—but Hadoop can also access Ceph via its POSIX IO interface, using ioctl calls for data location information. (6) Since Ceph is designed to serve as a general purpose file system (e.g. it provides a Linux kernel client so Ceph file systems can be mounted), if it supported Hadoop workloads well, it could also be a general solution to other storage needs.

In a preliminary experiment we compared the run time of Hadoop/HDFS with Hadoop/Ceph (Hadoop using the POSIX IO interface and Ceph not providing locality information) on a 40-node cluster running the word-count workload. We observed very similar run times which is very encouraging to us. Our poster will show more comparisons using a number of other important map/reduce workloads, and how name server scalability of Hadoop/Ceph compares to Hadoop/HDFS.

[1] Amazon. Simple storage service - developer guide (api version 2006-03-01). Web Page. docs.amazonwebservices.com/AmazonS3/2006-03-01/, March 2006.

[2] R. Ananthanarayanan, K. Gupta, P. Pandey, H. Pucha, P. Sarkar, M. Shah, and R. Tewari. Cloud analytics: Do we really need to reinvent the storage stack? In *HotCloud'09*, San Diego, CA, June 15 2009.

[3] Hadoop Project. Hadoop cluster setup. Web Page. hadoop.apache.org/core/docs/current/cluster_setup.html.

[4] W. Tantisiriroj, S. Patil, and G. Gibson. Data-intensive file systems for internet services: A rose by any other name ... Technical Report CMU-PDL-08-114, Parallel Data Laboratory, CMU, Pittsburgh, PA, October 2008.

[5] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn. Ceph: A scalable, high-performance distributed file system. In *OSDI 2006*, Seattle, WA, Nov. 2006.

[6] S. A. Weil, S. A. Brandt, E. L. Miller, and C. Maltzahn. CRUSH: Controlled, scalable, decentralized placement of replicated data. In *SC 2006*, Tampa, FL, Nov. 2006. ACM.

[7] S. A. Weil, A. Leung, S. A. Brandt, and C. Maltzahn. Rados: A fast, scalable, and reliable storage service for petabyte-scale storage clusters. In *PDSW 2007*, Reno, NV, November 2007.

[8] S. A. Weil, K. T. Pollack, S. A. Brandt, and E. L. Miller. Dynamic metadata management for petabyte-scale file systems. In *SC 2004*, Pittsburgh, PA, Nov. 2004. ACM.

[9] T. White. The small files problem. Web Page. www.cloudera.com/blog/2009/02/02/the-small-files-problem/.