Dynamic Load Balancing in Ceph

Esteban Molina-Estolano, Carlos Maltzahn, Scott Brandt, *University of California, Santa Cruz*

February 21, 2008

The Ceph distributed object-based storage system, developed at UC Santa Cruz, [1] uses CRUSH, a pseudo-random placement function, to decide which OSDs to store data on, instead of storing the placement information in a table. This technique offers multiple advantages. In particular, the amount of metadata stored per file is drastically reduced, reducing the load on the metadata servers and speeding up metadata accesses and clients need communicate with the metadata servers only for metadata operations, since they can directly calculate the correct data placement for read and write operations. [2] However, pseudorandom placement also brings challenges for load balancing, since data cannot be arbitrarily moved to other nodes.

We identify two types of load imbalance: persistent imbalance and transient imbalance. Persistent imbalance is caused by performance differences among nodes; we found that supposedly identical nodes in our cluster had up to four-fold differences in I/O performance. This can be addressed in Ceph by assigning different weights to different nodes in CRUSH.

Transient imbalance has two causes. First, a workload may be inherently imbalanced; for instance, a flash crowd on a single object can overload a storage node. Second, even without an imbalanced workload, transient imbalance may coincidentally occur: CRUSH's pseudorandom placement statistically distributes workloads well over time, but this does not guard against coincidental hotspots at any given moment.

We have a number of ideas for load-balancing techniques. We have added limited support for read shedding, where clients in a read flash crowd are redirected to replicas instead of the primary copy. This can be extended to allow clients to read from other clients in the flash crowd. We can switch primaries to distribute non-flash-crowd load from one primary to several primaries. We also have an algorithm to take a flash crowd of multiple writers to the same object and split the work among several nodes, by delaying synchronization.

We have tested the primary switching technique. When a single primary is overloaded by requests on multiple objects, those objects will typically have different sets of replicas. For each object, we temporarily transfer the primary role to one of the replicas. We have shown that primary shifting successfully relieves the load on the original primary and distributes it among several new primaries. However, we have not yet found workloads that saturate the primary in such a way that the load balancing makes the workload complete more quickly.

Future work includes testing the other techniques; characterizing which techniques speed up which workloads; and dynamically detecting overload to automatically invoke these load-balancing techniques.

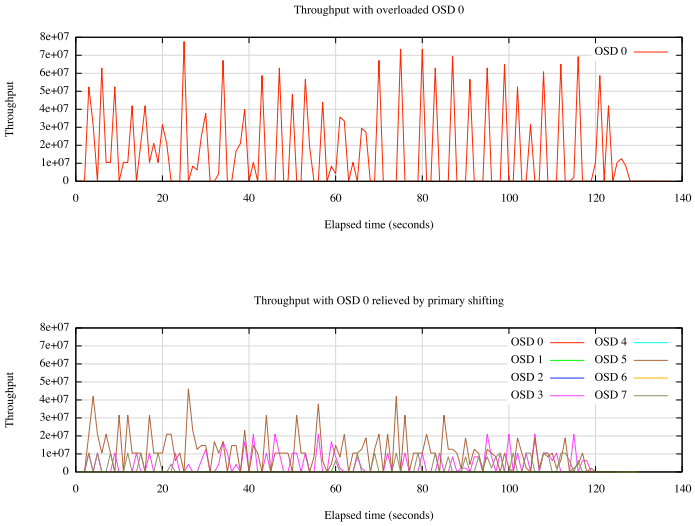

Figure 1: In this workload, eight clients write 256 MB each, and all writes initially have OSD 0 as the primary. The top graph shows the load on OSD 0. In the bottom graph, primary switching is activated, distributing the load among the OSDs.

References

[1] Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Carlos Maltzahn. Ceph: a scalable, high-performance distributed file system. In *USENIX'06: Proceedings of the 7th conference on USENIX Symposium on Operating Systems Design and Implementation*, pages 22–22, Berkeley, CA, USA, 2006. USENIX Association.

[2] Sage A. Weil, Scott A. Brandt, Ethan L. Miller, and Carlos Maltzahn. Crush: controlled, scalable, decentralized placement of replicated data. In *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, page 122, New York, NY, USA, 2006. ACM.