GeneralStore: Declarative Programmable Storage

Peter Alvaro

ABSTRACT

Despite the broad and growing diversity of storage-intensive applications, storage interfaces have evolved little over time. Fears of vendor lock-in and loss or inaccessibility of legacy data have discouraged the evolution of the POSIX API to better support existing and emerging workloads. This has long been a source of consternation for applications developers, who are often required to duplicate significant functionality to avoid inefficiencies [3].

Recently, the development of programmable storage systems (PSS) [5] has promised to upend this tradition, exposing a variety of composable subsystems as building blocks rather than a narrow storage interface. Such systems provide infrastructure programmers with great flexibility in implementing application-specific semantics while reusing trusted components. Unfortunately, the composition of subsystems is a low-level task that couples and obscures orthogonal concerns including functional correctness, reliability and performance. Building a new interface typically requires thousands of lines of carefully-written C++ code, an effort that must be repeated frequently as device and subsystem characteristics change.

The problem of programmable storage closely parallels the advent of the relational model for data management systems. The rate of evolution of the storage subsystem (due in part to advances such as solid-state devices and non-volatile memory) has far outstripped the rate of change of application code – an ideal setting for a "data independent" solution. However, the analogy only carries so far. The space of "physical designs" for a particular application implemented over a PSS, including both access methods and tunable parameters for low-level storage interfaces is much broader and more varied than in the relational setting. For example, a physical design must decide whether to use block-oriented or key-value interfaces, to use spinning disks, SSDs or some mix, to choose appropriate "tunables" such as block size (Ceph has 994 tunable parameters), as well as to implement and maintain secondary structures such as indexes or caches (or materialized views). At the same time, a language like SQL lacks the expressivity required to implement many of the applications that can be built atop a PSS.

In this abstract, we advocate a declarative approach to programmable object storage systems that leverages and integrates the hard-won lessons from the data management and storage communities. We propose GeneralStore, a programmable storage system vision that separates the concerns of functionality, performance and reliability, enabling a wide variety of optimizations without risking years of investment into code hardening (investments that are commonly made obsolete by the evolution of the storage substrate). As a proof of concept, we implement the CORFU protocol [1] atop the Ceph object storage system [4] using BRADOS, a declarative language that borrows syntax from Bloom [2]. We show how BRADOS is expressive enough to implement a succinct and intuitive shared log abstraction, yet "declarative" enough to give rise to a large set of physical implementations that can be costed and searched whenever physical storage characteristics change, whether due to software changes or hardware upgrades. We present a high-level architecture for GeneralStore that builds on the state of the art in data management and storage systems research, identifying difficult challenges and synergies for the research community.

1. COLLABORATORS

This abstract describes joint work with Noah Watkins, Michael Sevilla, Ivo Jimenez, Shel Finkelstein and Carlos Maltzahn.

2. REFERENCES

[1] M. Balakrishnan, D. Malkhi, V. Prabhakaran, T. Wobber, M. Wei, and J. D. Davis. CORFU: A Shared Log Design for Flash Clusters. NSDI'12.

[2] Bloom programming language. `http://www.bloom-lang.org`.

[3] M. Stonebraker. Operating System Support for Database Management. *Commun. ACM.*

[4] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn. Ceph: A Scalable, High-performance Distributed File System. OSDI '06.

[5] S. A. Weil, A. W. Leung, S. A. Brandt, and C. Maltzahn. RADOS: a scalable, reliable storage service for petabyte-scale storage clusters. In *PDSW '07.*

7^{th} *Biennial Conference on Innovative Data Systems Research (CIDR '17)* January 4-7, 2017, Asilomar, California, USA.