S4: A Simple Storage Service for Sciences

Matei Ripeanu[*] Adriana Iamnitchi[#]

[*]*Electrical and Computer Engineering, University of British Columbia*
2356 Main Mall Vancouver, BC V6T 1Z4, Canada
matei@ece.ubc.ca
[#]*Computer Science and Engineering, University of South Florida*
Tampa, FL, 33620-5399
anda@cse.usf.edu

***Abstract*— Amazon's Simple Storage Service (S3) is a storage utility that bundles at a single pricing point three storage system characteristics: high data durability, high availability, and high-speed data access. Many applications, and scientific applications in particular, do not need all these three characteristics bundled together. We argue that unbundling offers an opportunity to provide acceptably-priced storage and that future storage utilities designed to support science communities can offer multiple classes of service without introducing hidden, complexity-related costs.**

I. Introduction

Amazon's Simple Storage Service (S3) has attracted significant attention and a large user base due to a simple payment scheme (pay-as-you-go), unlimited storage capacity, open protocols, and a simple API for easy integration with applications. Yet, the current S3 design (and that of other existing storage utilities) does not satisfy a class of large storage consumers: the scientific communities. The incompatibility between scientific usage requirements and the service provided by S3 is caused by one design decision: bundling all utility storage characteristics to offer a single class of service and a single pricing point.

In this article we argue for the need to *unbundle* the performance characteristics of utility storage into multiple levels of service for durability, availability, and access performance. While this need has been discussed before for storage [1], it has not gathered traction and it has never been implemented apart from ad-hoc manual solutions. The timing for this discussion is better now than ever because:

- Storage utilities have proven successful (e.g., today S3 is capacity limited [2], other companies such as Microsoft, Google and Yahoo are rumored to plan similar offers) and continue to evolve rapidly. Recently, S3 offered a new charging scheme that favors intensive data usage. Reinvigorating now the discussion on optimal storage utility design will affect the design of new storage utility services targeted towards science community needs and will impact the evolution of existing services.
- Computing centers have changed focus from supporting isolated projects to supporting entire communities of users associated with a common scientific goal (e.g., TeraGrid). Providing storage as a utility is more apt to support collaboration and integration with applications.
- Data-intensive scientific applications continue to spur-in new production-mode collaborations. As such, economies of scale become a significant incentive both for users and providers of storage services to adopt the utility model.

The rest of this paper makes the case for unbundled storage supported by quantitative analysis based on Amazon's S3 and a real data-sharing science collaboration (DZero), and discusses the main challenges in designing such a service.

II. Why One-Size-Fits-All Does Not Work

Utility services should provide comparable performance with in-house services but at lower costs (by exploiting the economy of scale). Yet, today, for science collaborations, S3 proves to be both more expensive and offer lower performance than an in-house service. We believe that unbundling can reduce the costs of providing storage services by better responding to the characteristics of storage access for science applications.

We have analyzed [3] over 27 months of data access traces from DZero [4], a high-energy physics collaboration, to estimate its storage costs if it were to use Amazon's S3. The intense data usage (over 375 TB of stored data and 5.2PB of data processed during our trace collection interval) make using S3 prohibitively expensive. With today's S3 pricing scheme (roughly of $0.15/month/GB stored and $0.13—0.18/GB of data transferred) the monthly storage costs are above $80,000/month. A number of observations can help reduce this cost to less than a quarter: DZero data is 'hot' only for a relatively short period. For example, 40% of the files are used for no more than a week during the 27 months of traces and the median file lifetime is one month. 'Cold' data can be thus archived on cheaper, lower-performance storage without significant impact on application performance. Additionally, hot data can be cached near clients demanding it (either by S3 or by the application), thus reducing access costs. Similarly, derived data does not need costly long-term durability, as it can always be recomputed from primary raw data (often at lower costs). All these techniques, that could significantly reduce

storage costs, would be enabled if S3 provided a few classes of services and differentiated pricing.

To summarize: S3 bundles at a single pricing point three storage system characteristics: high durability, high availability, and fast data access. Many applications, however, do not need all these three characteristics bundled together – thus unbundling offers an opportunity to offer storage acceptably priced.

III. Is Unbundling Feasible?

The rest of this paper (1) presents additional arguments that unbundling can reduce costs for specific classes of applications; (2) argues that it is technically feasible, and (3) that the resulting system can remain simple, and thus usable.

We argue that unbundling can reduce the cost of offering storage for a significant set of applications/services. As Table 1 shows, each of the three salient properties that characterize a storage system (data durability, availability, and access-performance) requires different resources and engineering techniques. Thus, unbundled storage that offers reduced performance on some of these properties and uses only some of these resources can be provided at lower costs by using only a part of these resources/techniques.

Characteristics	***Resources and techniques to provide them***
High-performance data access	Geographical data (or storage) replication to improve access locality, high-speed storage, fat networks.
Durability	Data replication - possible at various scales: RAID, multiple locations, multiple media; erasure codes.
Availability	Server/service replication, hot-swap technologies, multi-hosting, techniques to increase availability for auxiliary services (e.g., authentication, access control)

Table 1: Different resources are needed to provide high performance data access, high data availability and long data durability are different

Unbundling can be easily exploited: a significant set of services requires storage that offers high-performance only on one or two of the above performance directions (Table 2). For example, an archival storage service puts a premium on durability but applications can survive with lower availability and access performance. In the case study we considered, DZero, the large share of data that is infrequently used could be well stored on tape to reduce costs. Similarly, distributed data caching demands fast access but not high durability.

Application class	***Durability***	***Availability***	***High access speed***
Cache	No	Depends	Yes
Long-term archival	Yes	No	No
Online production	No	Yes	Yes
Batch production	No	No	Yes

Table 2: Application classes and their associated requirements.

Additionally, in scientific communities, data storage requirements depend also on type of data: raw/derived. For example, derived data can be produced as necessary based on archived raw data. Users should be allowed to choose the best cost tradeoff between storing derived data on highly durable storage and using cheaper, less reliable storage but accounting for the possibility that data can be lost and recomputed as necessary.

Thus, the fact that each of the characteristics listed in Table 1 above requires different resources and techniques can be exploited to lower the cost of providing a service with characteristics targeted to each application. A solution that would address these requirements would necessarily include two key elements: *the ability to efficiently provide service in different classes* and maintaining *the usability* of the system. We discuss each of these requirements in turn.

Ability to provide service in different classes: Although there has been significant effort to provide highly reliable, highly available storage that can be accessed remotely, little research [7] has been done assuming these characteristics can be unbundled. Among the challenging problems that such an approach brings are defining a meaningful, adaptive and yet manageable set of service classes, providing efficient resource provisioning tools to offload peaks in demand between different service classes, and delivering quality of service guarantees. In particular, a competitive price for the service consumer requires efficient utilization of resources at the service provider's side. Consequently, dedicated resources for each performance characteristic or service class might be preferable for simplicity, yet they do not guarantee low costs.

The resulting system is easy to use: There are two main stakeholders in the resulting system: the service provider and the clients. From the point of view of the storage service provider, a number of service management questions are nontrivial (some of them are business related and require responses more related to game theory): Assuming a set of existing resources, how does the organization define the service classes and allocates resources to them so that it minimizes cost (and without impacting demand)? Can one easily transfer resources from one service class to another? Can the user easily transfer data from one service class to another? How does one do provisioning and capacity planning?

From the perspective of the client, the usability questions are relatively simpler: A limited number of service classes offered by the provider (rather than user-specified levels of service) will reduce complexity significantly. Additionally, these service classes should be configured to map well on well understood usage scenarios (archival, caching, etc.).

Additionally, to reduce complexity, the system may facilitate the ability to exploit cost reductions based on the specific usage patterns of science applications. For example, a storage utility for sciences may provide tools for automate, trace-based decision to identify cold vs. hot data (which leads to changing its service class); or tools to predict future data usage, based on well-agreed general phenomena, such as time-locality.

References

1. Sandeep Uttamchandani, et al. Polus: Growing Storage QoS Management Beyond a "4-Year Old Kid". in 3rd USENIX Conference on File and Storage Technologies (FAST'04). 2004. San Francisco, CA.
2. H. Havenstein, Web 2.0: Demand Strains Amazon Web Services, in PC World. 2007.
3. M. Palankar, et al. Amazon S3 for Science Grids: a Viable Solution? in submitted. 2007.
4. H. Zhang, A. Goel, and R. Govindan, Using the Small-World Model to Improve Freenet Performance, in Infocom. 2002.
5. Amazon Web Services. http:/s3.amazonaws.com,
6. Amazon, Amazon Web Services. http:/s3.amazonaws.com. 2007, Amazon Inc.
7. Efficient Replica Maintenance for Distributed Storage Systems, Byung-Gon Chun, Frank Dabek, Andreas Haeberlen, Emil Sit, Hakim Weatherspoon, M. Frans Kaashoek, John Kubiatowicz, Robert Morris, NSDI'06