Amazon S3 for Science Grids: a Viable Solution?

Mayur Palankar
Adriana Iamnitchi
Computer Science and Engineering
University of South Florida,
Tampa, FL 33620, USA
{mpalanka, anda}@cse.usf.edu

Matei Ripeanu
Electrical and Computer Engineering
University of British Columbia,
Vancouver, BC V6T 1Z4, Canada
matei@ece.ubc.ca

Simson Garfinkel
Center for Research on Computation and Society
Harvard University,
Cambridge, MA, USA
simsong@acm.org

ABSTRACT

Amazon.com has introduced the Simple Storage Service (S3), a commodity-priced storage utility. S3 aims to provide storage as a low-cost, highly available service, with a simple 'pay-as-you-go' charging model. This article makes three contributions. First, we evaluate S3's ability to provide storage support to large-scale science projects from a cost, availability, and performance perspective. Second, we identify a set of additional functionalities that storage services targeting data-intensive science applications should support. Third, we propose unbundling the success metrics for storage utility performance as a solution, to reduce storage costs.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software – *performance evaluation (efficiency and effectiveness).*

General Terms

Measurement, Performance, Experimentation.

Keywords

Utility computing, Performance evaluation, Amazon Simple Storage Service.

1. INTRODUCTION

Data Intensive scientific collaborations produce large amounts of data. Modern high-energy physics experiments, such as DZero [1], LHC [2], or SLAC [3], typically generate more than one TeraByte (TB) of data per day and may soon produce ten times as much [4]. Managing this amount of data requires significant human and material resources, resulting in corresponding high storage and management costs.

Amazon Web Services, a wholly owned subsidiary of Amazon.com, now offers the Simple Storage Service (S3) [5], a novel storage utility with a simple 'pay-as-you-go' charging model. Amazon claims its service offers infinite storage capacity, infinite data durability, 99.99% availability, and good data access performance [5]. S3 uses open protocols and provides sample source code allowing developers to easily integrate it into their existing applications.

This paper evaluates whether S3 is a feasible and cost effective alternative for offloading storage from in house maintained mass storage systems for today's scientific collaborations like DZero, LHC, or SLAC. To this end, we characterize S3's observed availability and data access performance using a collection of our own nodes and geographically-distributed PlanetLab nodes [6]. We use this characterization in conjunction with more than two years of real traces from a scientific community, the DZero Experiment, a high energy physics collaboration that spans 18 countries and has more than 500 active users. We evaluate the feasibility, performance, and costs of a hypothetical S3—supported DZero collaboration.

The contributions of this paper are:

- The first independent characterization of S3 in terms of data user-observed performance.
- An evaluation of the costs of outsourcing the storage functions to S3 to support a data-intensive scientific application and an analysis of alternatives to reduce costs.
- A discussion of S3 functionality and security features, and recommendations for improvement in the context of data-intensive collaborative applications.

The rest of this paper is organized as follows: Section 2 gives an overview of S3's core concepts, architecture, and functionality. Section 3 presents data usage characteristics in science grids and looks at storage service requirements for such applications. Section 4 presents a measurement-based evaluation of S3 performance. Section 5 estimates the S3 costs and discusses various models for using S3 to support DZero-like applications. Section 6 lists out future discussion topics and suggestions for improving S3. Section 7 summarizes our study and outlines future research directions.

2. AMAZON S3

S3 is supported by a large number of computer systems distributed across multiple data centers [7] in the United States

DADC'08, June 24, 2008, Boston, Massachusetts, USA.

and Europe (S3-Europe) and is expected [5] to offer low data access latency, infinite data durability, and 99.99% availability (S3 pays users for average monthly uptime < 99.99%). Since its launch, S3 has acquired a large user base ranging from home users and small businesses to large business enterprises [8]. Currently, S3 stores over 5 billion user objects and handles over 900 million user requests a day [9].

In addition to S3, Amazon Web Services offers to sell virtual computer time at the cost of $0.10 per CPU hour on its Elastic Compute Cloud (EC2). The primary relevance of EC2 to this paper is that there are no bandwidth charges for data sent between EC2 and S3 (except between EC2 and S3-Europe, which is charged). As a result, scientific data stored on S3 can be cheaply processed using virtual EC2 hosts.

2.1 Concepts and Architecture

Data stored in S3 is organized over a two-level namespace. At the top level are buckets–similar to folders or containers–which have a unique global name and serve several purposes: they allow users to organize their data; they identify the user to be charged for storage and data transfers, and they serve as the unit of aggregation for audit reports. Each Amazon Web Services (AWS) account may have up to 100 S3 buckets.

Each bucket can store an unlimited number of data objects. Each object has a name, an opaque blob of data (of up to 5GB), and metadata consisting of a small set of predefined entries and up to 4KB of user-specified name/value pairs.

Users can create, modify and read objects in buckets, subject to access control restrictions described in the next section. Renaming an object or moving it to a different bucket requires downloading the entire object under one name and writing it back to S3 with the new name. Search is limited to queries based on the object's name and to a single bucket. No metadata or content-based search capabilities are provided.

Charging for the S3 service is based on storage volume (currently at a rate of $0.15/GB/month; $0.18/GB/moth if stored in Europe), data transfer activity (at $0.10/GB for uploads and between $0.13/GB and $0.18/GB for downloads), and a per-transaction charge ($0.01 per 1,000 PUT and LIST operations; $0.001 for each GET operation; DELETE is free). Regardless of the owner of an object or the identity of the user accessing the object, all charges are directed to the owner of the bucket that stores the object generating the charges.

2.2 The Security Model

When users register with Amazon's Web Services, they are assigned an identity and a private key. Both keys are permanently stored at Amazon and can be downloaded from the Amazon's Web Services website. Clients authenticate using a public/private key scheme and keyed-hash message authentication code (HMAC [10]). Because the private key is made by Amazon and downloaded from the website, the security provided by S3 is equivalent to security provided by a simple password; this password can be reset by anyone who can receive email at a registered email address. Each S3 account is linked to a credit card used for account billing.

Access control is specified using access control lists (ACL) at the granularity of buckets or objects. Each ACL can specify the access attributes for up to 100 identities. A limited number of access control attributes are supported: read for buckets or objects, write for buckets only, and read and write ACL. Buckets can be configured to store access log records for audit purposes. These logs contain details such as the request type, the object accessed, and the time the request was processed.

2.3 Data Access Protocols

Currently, S3 supports three data access protocols: SOAP [11], REST [12] and BitTorrent [13]. Of these, BitTorrent deserves special attention. BitTorrent is a popular file-sharing protocol that enables efficient cooperative data distribution: data is initially distributed at one or more seed sites that are pointed to by a tracker. As clients begin to download a BitTorrent file, those clients register themselves with the tracker and make portions that they have downloaded available to other clients. S3 can provide both tracker and seed functionality, allowing for substantial bandwidth savings if multiple concurrent clients demand the same set of objects.

3. CHARACTERISTICS OF SCIENCE GRIDS

Data produced, stored and used in science grids have particular scale and usage characteristics. This section surveys the usage characteristics of data intensive scientific collaborations and their implied requirements on the storage infrastructure. To quantify this characterization we focus the discussion on DZero [1], a representative high-energy physics collaboration that processes data generated by the particle accelerator at Fermi National Accelerator Laboratory.

3.1 Data Usage Characteristics

Particular to scientific communities is the intense usage of data: jobs submitted by hundreds of users process massive collections (TeraBytes), organized in hundreds to thousands of GB-sized files. For example, the 113,062 jobs submitted by the 561 world-wide located DZero scientists processed more than 5.2 PetaBytes of data between January 2003 and March 2005, a sustained access rate of over 78 MBps. The 5.2 PB of processed data occupies 375 TB of storage organized in almost one million distinct files [14]. At the same time, access to data can be shared by tens or even hundreds of users: in DZero files are accessed by 45 different users.

Another characteristic is co-usage: in science collaborations groups of files are often used together. Taking the high-energy physics project DZero as a case study again, each data analysis job accessed on average 102 files, with a maximum of more than 20,000 files. The need for simultaneous access to multiple files stresses the problems brought up by the large file size, requesting transfers of data collections in the order of TeraBytes. For example, the largest 10 datasets in the DZero traces analyzed in [15] are between 11 and 62 TB.

Finally, a significant part of the data, the so-called derived data, can be recomputed from raw data which is typically generated by scientific instrument (e.g., an astronomical observatory). This particularity allows for flexibility in data management solutions by trading data storage and transfer costs for computation costs: it sometimes may be more efficient to regenerate derived data than to store it or transfer it between remote locations.

3.2 Storage Service Requirements for Data Intensive Scientific Applications

A storage infrastructure targeting data intensive scientific communities must provide:

- *Data durability*: Depending on the specifics of each science project, losing experimental (raw) data may be costly (since repeating a physics experiment may require operating expensive instruments) or even unacceptable. This results in strong durability requirements for raw data. However, derived data can generally be reconstructed from raw data at the cost of additional computation.

- *Data availability*: Data availability quantifies the successful access to previously stored data. Although most of the data is used for batch computations that do not require high availability by themselves, the fact that these operations often require co allocation of expensive resources (e.g., large compute resources, visualization equipment) increases the premium put on availability. Note that durability does not imply availability – data can be stored and eventually be accessible but not available at the time of the request. However, availability requires durability. Finally, service availability for uploads rather than retrieval is important since data can be temporarily stored at experimental facilities only for limited periods of time.

- *Access performance*: While data archival is an important use case, we expect that the predominant use case in our context is live remote storage. Thus, fast data access is a key to support science applications.

- *Usability*: Although ease of use can be quantified across multiple directions, the main characteristic of interest in the context of this paper is a set of protocols and APIs that allow composability with higher-level services for easy integration with science applications.

- *Support for security and privacy*: Science applications are often collaborative with complex data sharing arrangements between multiple parties, users or institutions. The security infrastructure should enable defining and enforcing such sharing arrangements.

- *Low cost*: Ultimately, cost is the main driver for adopting the storage utility paradigm. Utilities have the potential to benefit from economies of scale and we believe data storage is ripe for such development. However, lack of standardization and the particular requirements of data intensive science might delay or make inopportune the adoption of storage utilities.

4. AMAZON S3 EVALUATION

Our evaluation of S3 is driven by the requirements of data-intensive scientific applications outlined in the previous section. This section describes our experimental setup and presents our quantitative evaluation of S3.

4.1 Experiment Setup

We conduct three different sets of experiments in order to evaluate the services provided by S3 using the Java, Python, and C++ implementations of the S3 REST API. Our choice for REST is motivated by its performance (REST is faster than SOAP because the data is sent raw without the need to encode using BASE64) and popularity (85% of current S3 usage is based on the REST protocol [16]).

We used five nodes on the public Internet for our experiments: four PlanetLab nodes and one dedicated machine at the University of South Florida (USF) in Tampa, Florida. We chose the PlanetLab nodes at locations that geographically approximate DZero user location: hence, two nodes were located in Europe (one in Germany and one in France) and two were located in the US (in New York and California).

We also conducted a series of measurements from several servers located inside Amazon's EC2 cloud. Because both S3 and EC2 are within Amazon's border routers, these experiments were able to measure the availability and performance of S3 free from the impact of Internet congestion and disruptions.

4.2 Data Durability

While characterizing S3 data durability is not the main goal of our study, we mention that, during the twelve-month time span of running S3 experiments, we have not observed even a single case of permanent data loss. An experimental study of different magnitude is required to characterize the data durability offered by the S3 service. Constrained by limited budget of experimenting with the service, we limited the size of the files used for our experiments to 1GB. We used more than 10,000 files for various experiments with file sizes ranging from one Byte to 1GB. Due to the limited budget, we also had to watch the number of accesses to large file sizes. More than 137,000 requests to and from the S3 server were made during the period of study.

4.3 Data Availability

Between March 20th and May 1st 2007 we conducted a series of tests from Amazon's EC2 cluster to determine availability and bandwidth of the S3 offering. Before the start of the test we stocked an S3 bucket with a variety of test objects with sizes of 1KByte, 1MByte, 16MBytes, and 100Mbytes (stored at S3 and not S3-Europe). The tests consisted of randomly writing objects of these sizes and reading the pre-stocked objects. The probes were separated in time by a random delay where the length of the delay followed a Poisson distribution, so that the statistical properties of the samples would have the same statistical properties of the system being measured (see [17] for a discussion of this technique). Our system retried each request a maximum of 5 times, and then declared a failure.

Observed availability from EC2 was quite high. In a total of 107,556 tests (each consisting of a read and write) from EC2, we encountered 5 instances where a HTTP PUT request needed to be retried because S3 indicated that the PUT had failed (S3 returns the MD5 hash of objects that are successfully PUT; in these cases it did not return an MD5 hash), and 23 cases in which the HTTP PUT request timed out and no response code was returned. Reads were more reliable: there were only 4 cases in which an HTTP GET had to be retried once, and 1 case in which it had to be retried twice.

Amazon recommends retrying failed access attempts with an exponential back-off. Of course, if the software never declares a failure, then 100% availability will always be observed unless data is lost or the S3 service is discontinued. Restricting the analysis to the 19,630 operations of 1MByte or greater, we found zero probes where the throughput for write requests was less than 10KB/s and just 6 (0.03%) where the throughput was less than 100 KB/s.

We also ran a series of S3 availability tests from a dedicated machine at USF. During this trial we repeatedly downloaded the same object every 15 minutes, for a period of 23 weeks and a total of 15,456 access requests. We observed an availability rate of 95.89% after the original download attempt, 98.06% after the first retry, 99.01% after the second retry, 99.76% after the third retry, 99.94% after the fourth retry and a full 100% availability after the fifth retry. Unlike our tests from EC2, these tests were influenced by both S3's internal systems and by Internet connectivity between Amazon and USF (Note, however, that for the numbers we report above we have eliminated all test cases where we have been able to detect a network fault by immediately after our unsuccessful attempt to access S3 data.)

4.4 Data Access Performance

Our objective in this section is to evaluate the access performance to S3-stored data from a client perspective. We compare the download time for objects sized 1B, 1KB, 1MB, 16MB and 100MB, with a total of 13,178 downloads.

4.4.1 Single-threaded performance

Figure 1 presents the cumulative distribution of observed bandwidth for reads from EC2 to S3. Each trace corresponds to a different object size, with 1Byte objects on the left and 100MByte objects on the right. The time for downloading the 1 Byte object is dominated by the S3 transaction overhead; the plot indicates that S3 can sustain a maximum of 100 transactions per second, with an average of roughly 50 transactions per second. The other end of the scale indicates a maximum bandwidth of approximately 21 MB per second, with average peak bandwidth of approximately 17 MB/sec.

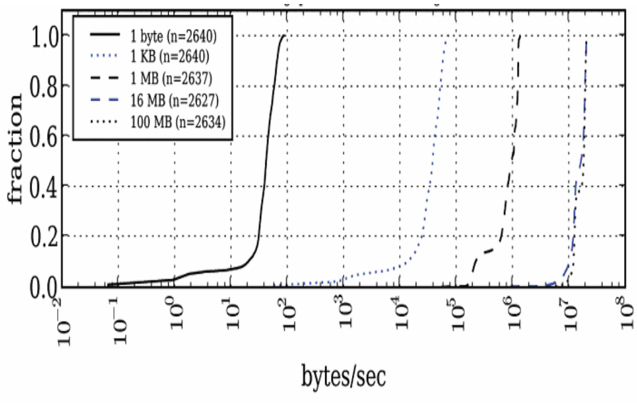

Fig 1. Cumulative Fraction graphs showing time to access S3 objects of 1B, 1KB, 1MB, 16MB and 100MB from EC2.

Amazon states that S3 was designed to store large objects; our experience confirms this. The performance of reading even 1MB-sized objects suffered due to transaction overhead. Only when we were reading objects of 16MB and larger did our tests enjoy consistently high performance.

We repeated all experiments to estimate the time to upload data to S3. We find that the upload time is largely similar to the download time for each location, with the exception that 1MB-sized objects are much faster to write than to read-back, probably a result of write-caching.

4.4.2 Concurrent performance

To simulate a number of experimenters attempting to access the same scientific data stored in S3, we performed a series of tests in which two virtual machines attempted to repeatedly access the same data stored in the same bucket. In our experiment one virtual machine ran on the EC2 cluster usma1 and the second running on cluster usma2, accessed the same bucket with repeated 100MB GET and PUT operations. The virtual machines were coordinated, with each executing 1 thread for 10 minutes, then 2 threads, then 3 threads, and so on up to 6 threads, at which point the experiment reset back to 1 thread. This experiment was repeated for 11 hours.

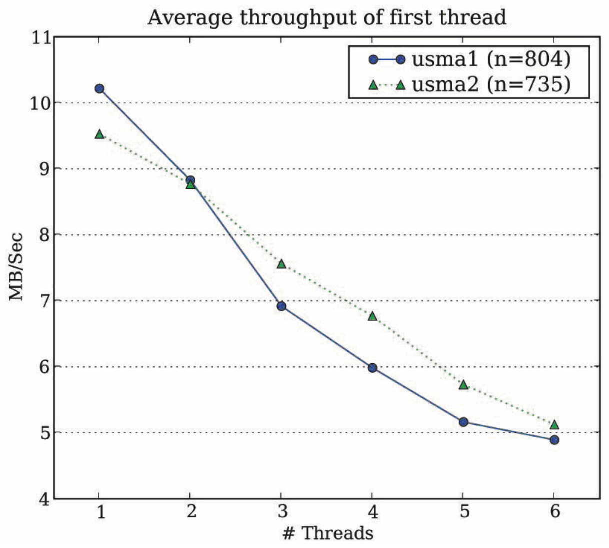

Fig 2. Performance of 100MB GETs from S3 for one thread as the number of concurrent threads running on the same virtual machine instance increases.

As shown in Figure 2 and 3, as the number of threads increased the per-thread bandwidth decreased but the aggregate bandwidth increased. With two machines running with six threads each, we enjoyed a bandwidth of roughly 30 megabytes per second, which is very close to the maximum bandwidth (250 Mbits/sec, or 31 Mbytes/sec) that Amazon states is available to any single S3 object or EC2 instance.

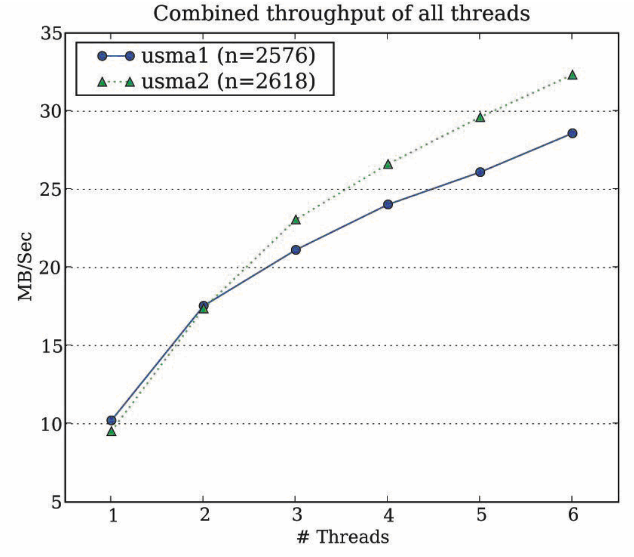

Fig 3. Performance of 100MB GETs from S3 for combined threads (down) as the number of concurrent threads running on the same virtual machine instance increases.

4.4.3 Remote Access Performance

We compare the download time from five access locations (one dedicated node located at USF and four other PlanetLab nodes). The data set includes 28 experiments (4 times a day over 7 days) to quantify variability due to competing Internet traffic. Our experiments confirm that the location of the client and the time of the day impact the observed access performance. Figure 4 presents the average, minimum and maximum download bandwidth at each node as a function of the size of file downloaded. Additional data is available in our technical report [18].

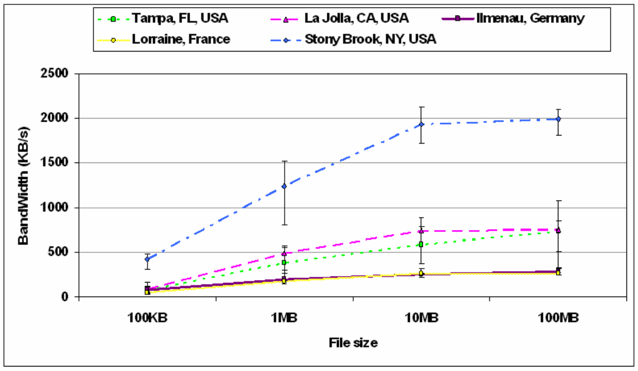

Fig 4. Average, minimum and maximum observed download bandwidth for different client locations and file sizes.

4.5 Downloading Files via BitTorrent

Typically, multiple replicas of the same data item are available simultaneously at various sites participating in a virtual organization. BitTorrent enables partial parallel downloads from each of these replicas thus allowing for faster downloads. For the science community, the availability of data access through BitTorrent protocols is relevant as it enables simple integration of cooperative caching mechanisms to improve access performance. Additionally, the cooperative cache created by BitTorrent is a direct solution to reduce S3's transfer charges while preserving the data durability and availability offered by S3.

The main goal of our experiment is, first, to compare the BitTorrent enabled data access performance with that offered by regular S3 transfers and, second, to understand the load balance between S3 and other data sources when multiple sources ('seeds' in BitTorrent parlance) are available.

To quantify the effectiveness of using BitTorrent, we download data from a single location (Tampa, FL) while varying the number of seeds hosted on PlanetLab nodes and always maintaining a seed at S3. We measure the download time and the amount of data downloaded from S3.

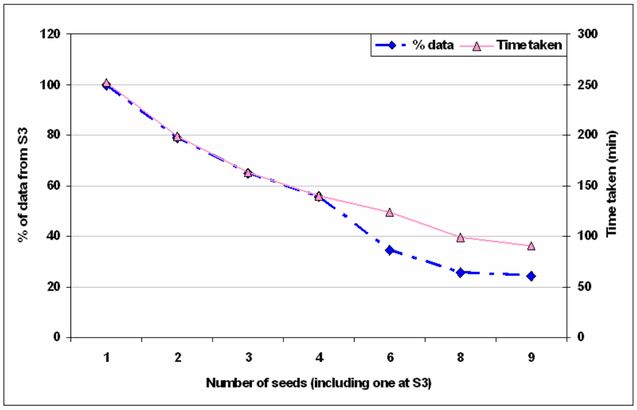

Fig 5. Time taken and percentage of data downloaded from S3 as a function of the number of seeds present in the system.

Figure 5 demonstrates that S3 contributes a large percentage of the data volume. To check for any unfair play by S3, experiments were repeated on dedicated machines at USF instead of PlanetLab. It appears that S3 is fair and its larger contribution in BitTorrent was due to unbalanced load and network conditions.

To continue with our black-box exploration, we tried to use the S3-provided BitTorrent tracker without incurring the expense of using S3 as a BitTorrent seed, thus benefiting from the high availability of the Amazon provided tracking service at a very low cost. We discovered we could do this by revoking the read permits from the S3-stored object (i.e., mark it as private). This approach blocked nodes from downloading data from S3 and forced the client instead to use other seeds. Our experiments show (Figure 6) that S3 permits this use; we observed only small performance degradation due to a lower parallelism when downloading data from fewer sources [24].

To summarize, our S3 experimental results show good availability, variable download time depending on the node location and file size, and a fair use of the BitTorrent protocol.

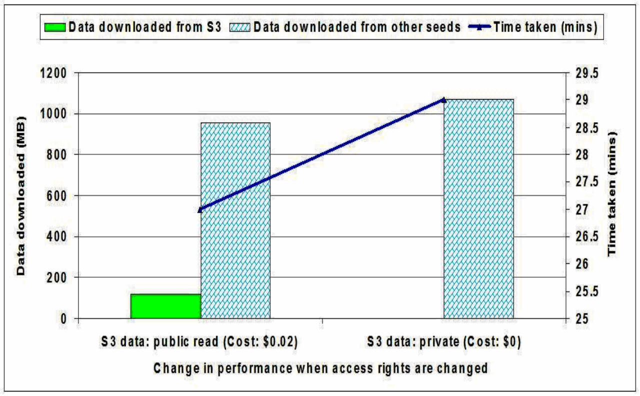

Fig 6. Marking the S3 seed as private (and in effect freeriding the S3 tracker) leads to minimal change in the download time (8% increase) while the Amazon related costs are almost entirely eliminated. The figure presents the volume of data downloaded from S3 (left bar) and five other seeds (right bars) and the total client download time in two scenarios: with S3 data public and private. Right-hand axis not scaled to zero.

5. AMAZON S3 FOR SCIENCE GRIDS

This section addresses three issues related to the feasibility of using S3 to support data intensive scientific applications: it estimates the cost of using S3, it evaluates whether S3 performance is adequate to support scientific collaborations, and it analyses S3 security functionality from the perspective of supporting the complex data-sharing required. While answering these questions in general is a generous subject, our approach is based on a case study: we use DZero-generated load and study different deployment scenarios that combine S3 and data caching.

5.1 Costs

While the cost of storage systems and their management has continued to decrease in recent years, it still represents a major expense, often the single most expensive item when running an IT infrastructure. In this section, we consider the purely hypothetical case of DZero's using S3 for its data needs. Two types of costs are thus to be considered: data access costs and data storage costs. DZero storage characterization as reported from 27 months of traces is summarized in Table 1.

Table 1. DZero trace characteristics

Trace recording interval	01/2003 – 03/2005
Number of jobs	113,062
Hours of computation	973,892
Total storage volume	375 TB
Total data processed	5.2 PB
Average data access rate	273 GB/hour

Assuming all DZero data is stored by and accessed from S3, the annual costs are $691,200 per year for storage ($829,440 for S3-Europe) and $335,012 per year for transfer (initial upload and download), a total of $1.02 million per year. Each of these costs can be reduced in various ways. In the rest of this section we first outline two approaches to reduce storage costs by exploiting access patterns and the fact that part of the data is derived, and then we outline approaches to reduce data transfer costs based on collaborative caching enabled using BitTorrent.

Storage costs can be reduced by archiving "cold" data on low-cost storage and maintaining only the data most likely to be used on high-availability, low-latency storage. An analysis on the DZero traces shows that a significant part of data is used for only limited time periods. This data can be archived on slower durable storage without an impact on performance. If we consider the lifetime of a file as the interval between the first and the last access to it as recorded in the 27 months of DZero traces, then about 30% of the files do not live longer than 24 hours, 40% not more than one week, 50% have a lifetime shorter than one month, while about 35% were still in use more than five months after the first recorded access. Consequently, out of the 4.54TB of data accessed each day, 30% will not be needed after 24 hours. S3 provides a log facility that can be used to determine which objects are suitable for archiving.

A second way to reduce storage costs is to only store raw data and derive the rest of the data from raw data. Our workloads do not contain sufficient information to allow us estimate the potential benefits of this approach.

Transfer costs can be reduced by using local caches. Data in DZero is highly cacheable: experimental evaluations [14] with various cache replacement algorithms performed on the DZero traces show that TB-sized local caches may reduce data transferred from the permanent storage to under 2.5% of the cache size. For example, for a cache of 50TB, the amount of data that needs to be transferred on average per job is 0.013% of the cache size, or 6.6GB [14]. Given that over the 27 months, 113,062 jobs were submitted within the subset of traces, the cost of cache misses leading to accesses to S3 is $.87 per job or $98,748 for all the jobs in our traces. This leads to reducing data transfer costs to $43,888 per year, 20 times lower than in the original scenario without caching[1]. BitTorrent could be used to further reduce transfer costs by building a large collaborative cache across DZero sites.

S3 data transfer costs can also be reduced by using computation close to where data is stored. Since data transfers between S3 (not S3-Europe) and Amazon's Elastic Computing Service (EC2) are not charged, EC2 hosted computations can be used to replace the data transfer costs for derived data with the cost to recompute this data on EC2 ($0.1/hour of computation). Over the period of our workloads, this result in an average of $43,284 per year (relatively close to the access costs above).

Our experience using EC2 was excellent. Although Amazon makes no guarantee regarding the durability or availability of any EC2 instance, we ran instances for a total of 6 CPU months and had only one case in which an instance was rebooted without our

[1] This discussion ignores the additional costs of purchasing and maintaining the caches which is low relative to the other costs presented here.

initiation. After reboot, the contents of our instances' hard drives were preserved, even though Amazon states that all information on an instance should be considered volatile and persistent data should be stored on S3.

Although the ideas above are acknowledged methods in data-intensive computing, they require support from the application side or from S3, as discussed further in Section 6.

5.2 Performance

A second question we are set to address is whether the potential savings obtained by outsourcing data storage come at the price of performance degradation.

5.2.1 Impact of variability of data access performance

We have noted that data access performance varies with the location of the downloading node. We suspect that S3 operates multiple datacenters (in both US and Europe locations) and makes data placement decisions depending on the location of the user creating the bucket that stores the data. This assumption, if true, explains the observed variability in download times that goes beyond the location of the downloading node and the time of day. It may also result in the need to change the data placement decisions to reduce the performance differential that different participants to scientific collaborations spread across the world might experience.

5.2.2 Impact of access performance to individual data items

First, we note that, for batch processing with little or no interactive user input, the relatively slow access S3 provides to individual data items does not have a significant impact on user observed performance as long as jobs are specified in advance and S3 is able to provide data at an overall rate faster than the rate at which it can be processed. To this end, an efficient system using S3-hosted data would only need limited local storage and a well-designed job management system that makes use of batch job information to proactively retrieve S3-stored data while jobs are still in compute queues. Where pre-fetching is not an option (e.g., for interactive computations) the access performance to individual data items may prove to be the driving factor for observed performance.

5.3 Subsequent Pages

The assessment of the security functionality should start with an evaluation of the risks involved. S3 presents the traditional risks that outsourced data storage systems present: permanent data loss, temporary data unavailability, loss of data confidentiality, and malicious data modifications are of concern. Some risks are mitigated by the security scheme S3 uses (e.g., transport privacy can be provided by using TLS) while others can be mitigated by user-level solutions, such as cryptographically signing data stored in S3. On the other hand, S3 provides no check-pointing or backup facility for recovering data that is accidentally erased or modified.

S3 charging scheme introduces an additional risk: direct monetary loss. This risk is magnified by the fact that S3 does not provide a solution to limit the amount users stand to lose in case of an attack. For example, an attacker could repeatedly transfer data to or from an S3 bucket to cause direct monetary loss to the owner of that bucket.

S3's security model has the important merit of being simple. Simplicity however, comes at the price of limited support for large, collaborative applications that aim to control access to data resources offered by multiple participants. We summarize these limitations below:

- *Crude access control scheme*: S3 access control solution based on access control lists does not scale well to large systems with thousands of users and million of entities. Additionally, S3 supports only a limited number of access rights (e.g., there is no write control at the individual object level but only at the bucket level) and the access control lists are limited in size (100 principals).
- *Lack of support for fine-grained delegation*: Even moderately complex data-intensive scientific collaborations today make use of delegation for efficient data management and processing. For example, users delegate access to specific datasets to programs that operate on their behalf on remote computers. Higher risks involved by science data on S3 (as a result of direct monetary loss risks) make proper access control delegation models even more necessary. S3, however, lacks any support for delegation which is a major obstacle to adopt S3 for large-scale science collaborations.
- *Implicit trust; no support for non-reputability*: The implicit assumption is that users trust S3 entirely, and thus S3 does not provide unforgeable ‘receipts’ for transactions (signed certificates) that a user can present to a third party to demonstrate that a specific data item had been stored. Additionally, the invocations recorded by S3 and presented in the audit trail are not digitally signed, either by the users or by Amazon. This makes the audit trail repudiable. Worse, given that Amazon security solution has the essential properties of shared secret scheme, it is impossible to provide non reputability.
- *Unlimited risk*: S3 does not offer any support for user-specified usage limits, (e.g., quotas per bucket owner or per delegated principal). As a result, the potential damage that an attacker (or, worse, simply a buggy program) can produce is limited only by the ability of the attacker to access S3 data and by the limit on the user’s credit card.

Some of these risks can be mitigated by using S3 as the backend of a storage system, and having users connect to a front-end running on Amazon's EC2 service. The front-end would be responsible for individual account management, fine-grained trust decisions, and billing.

6. RECOMMENDATIONS FOR NEXT GENERATION STORAGE UTILITY SERVICES

S3 has attracted a large user base due to its simple charging scheme, unlimited storage capacity, open protocols, and simple API for easy integration with applications. Yet, we believe that the current S3 design needs to be improved before it can provide durable storage. Nevertheless, it is important for the scientific community to discuss the applicability of utility storage systems such as S3 for three reasons.

First, storage utilities have proven popular for consumers and small businesses and continue to evolve rapidly. Reinvigorating the discussion on optimal storage utility design will affect the design of future storage utility services that target the needs of the science community.

Second, computing centers have changed focus from supporting isolated projects to supporting entire user communities grouped around common scientific goals (e.g., TeraGrid [19]). Providing storage as a utility is more apt to support user collaboration and integration with applications and fits the mandate of existing computing centers and large scale cyber infrastructure deployments.

Finally, data-intensive scientific applications continue to spur-in new production-mode collaborations. As such, economies of scale become a significant incentive for both storage consumers and providers to adopt the utility model.

The rest of this section discusses possible architectural and functionality changes to improve the suitability of S3 to support science applications.

6.1 Unbundling Performance Characteristics

Utility services should aim to provide comparable performance with in-house services and exploit economies of scale to reduce costs. Yet, we estimate that, today, while S3 may offer similar performance to an in-house service, it does not offer a compelling cost advantage.

We believe that the solution to reduce storage costs is to understand and respond to application requirements. More specifically, S3 bundles at a single price point three storage system characteristics: infinite data durability, high availability, and fast data access. Many applications, however, do not need all these three characteristics bundled together – thus 'unbundling' by offering multiple classes of service targeted for specific application needs may lead to reduced costs and thus lower utility prices. The rest of this section presents additional arguments that unbundling can reduce costs for specific classes of applications and argues that it is technically feasible.

Unbundling could reduce costs. As Table 2 shows, each of the three salient properties that characterize a storage system (data durability, availability, and access-performance) requires different resources and engineering techniques. For example, Amazon could offer a class of service that offers durability but data may be unavailable for up to 7 days every month; such a service might be cheaper for Amazon than its current offering.

Unbundling can be exploited by applications. A significant set of services requires storage that offers high-performance only on one or two of the above performance directions (Table 3). For example, an archival storage service that puts a premium on durability can survive limited periods where data is not available. In 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.

Table 2. The resources needed to provide high performance data access, high data availability and long data durability are different

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 levels: hardware (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 3. Application classes and their associated requirements

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

In scientific research the cost associated with losing data depends on whether the data is raw data or derived data obtained from raw data through processing. Derived data can always be reproduced as necessary based on archived raw data. Users could be allowed to choose the best cost tradeoff between storing derived data on highly durable storage and storing it on cheaper, less reliable storage that might result in data loss and the need to recomputed lost data as necessary.

We note that a possible argument against multiple classes of service is that they may introduce nontrivial management costs at the utility service provider (in terms of resource management, billing clients, and educating application developers to harness them). In fact, a similar argument has been made regarding the introduction of quality of service in the Internet [20] [21]. While we agree that these costs can be nontrivial, we believe that future research in this area can reduce these costs through automation and make differentiated storage services through unbundling and appealing alternative.

6.2 Increasing Flexibility

Our S3 experience allows us to recommend primitives that will extend the flexibility offered by novel storage infrastructures that cater to the demands of data-intensive science applications. These recommendations could also prove valuable to large Grid deployments like TeraGrid or WestGrid [22] which are moving towards offering infrastructure services for science, similar in goals to those offered by Amazon's S3 and EC2 to commercial applications.

6.2.1 Enhanced security functionality to support complex collaborations

Our analysis reveals that the simple security model offered by S3 does not support complex collaborations and exposes users to major risk. We believe that two key components are required: the ability to limit potential damage in the case of an attack and the support for fine-grained delegation.

6.2.2 Additional functionality for better usability

A number of additional service functionalities would significantly simplify integration with applications. Three main such functionalities we recommend are (1) metadata based searches; (2) ability to rename objects; and (3) ability to mutate access control lists [23].

7. SUMMARY

S3 was not designed for the science community. Indeed, the science community has specific requirements and extreme challenges regarding data usage. As such, this paper should be read not as a critique of S3, but as a set of recommendations to any storage provider that would like to serve this community.

The contributions of this paper are in evaluating S3 as a black box and in formulating recommendations for integrating S3 with science applications and for designing future storage utilities targeting this class of applications. Costs can be reduced by exploiting data usage and application characteristics to improve performance, and, more importantly, by introducing user-managed collaborative caching in the system. In effect, our recommendations are driven by S3 billing structure: we recommend using S3 for the costly tasks of providing high data availability and durability (where costs are driven up by specialized hardware and nontrivial engineering effort) and employ caching at the edges of the system to reduce the access volume when the usage patterns allow. These recommendations may not only reduce the S3 bill but will also significantly improve performance due to a cacheable workload specific to these collaborations.

We identify application requirements that are not currently satisfied by S3. While S3 successfully supports relatively simple scenarios (e.g., personal data backup) and can be easily integrated in the storage tier of a multi-tiered Web application, its existing security functionality is strikingly inadequate to support complex, collaborative environments like the ones in today's scientific collaborations. More precisely, S3 lacks in terms of flexible access control and support for delegation and auditing, and it makes implicit trust assumptions. This lack of functionality is troubling when direct financial loss is at stake.

Finally, we observe that S3 bundles at a single pricing point three storage system performance characteristics: infinite data durability, high availability, and fast data access. We believe that unbundling these characteristic by offering multiple classes of service targeted for specific application needs will reduce the storage utility price.

8. ACKNOWLEDGMENTS

We thank Ayodele Onibokun for his early efforts invested in this project and Abullah Garaibeh for feedback. We acknowledge the financial support for the Amazon service expenses from the Department of Computer Science and Engineering at University of South Florida and the Center for Research on Computation and Society at Harvard University's School of Engineering and Applied Sciences. We have used the DZero application traces gracefully provided by the DZero team at the Fermi National Laboratory.

9. REFERENCES

[1] The DZero Experiment. http://www-d0.fnal.gov

[2] The Large Hadron Collider. http://lcg.web.cern.ch/LCG

[3] The Stanford Linear Collider. http://www2.slac.stanford.edu/vvc/experiments/slc.html

[4] The Compact Muon Solenoid at CERN. http://cmsinfo.cern.ch

[5] Amazon Web Services. http:/s3.amazonaws.com

[6] PlanetLab Consortium. http://planet-lab.org

[7] Garfinkel, S. 2007. Commodity Grid Computing with Amazon's S3 and EC2, Login, USENIX, February 2007.

[8] Kirkpatrick, M. 2006. Amazon releases early info on S3 storage use, http://www.techcrunch.com/tag/s3, July 2006.

[9] Bezos, J. keynote talk at Web 2.0 Expo, April 2007.

[10] Krawczyk, H., Bellare, M., and Canetti, R. 1997. HMAC: Keyed-Hashing for Message Authentication, IETF - Network Working Group, 1997.

[11] W3C, Soap Version 1.2, June 2003, http://www.w3.org/TR/soap/

[12] Fielding, R. T. 2000. Architectural Styles and the Design of Network-Based Software Architectures, PhD Dissertation, University of California, Irvine, 2000.

[13] BitTorrent. http://www.bittorrent.com

[14] Doraimani, S. 2007. Filecules: A New Granularity for Resource Management in Grids, Masters Thesis, University of South Florida, April 2007.

[15] Iamnitchi, A., Doraimani, S., and Garzoglio, G. 2006. Filecules in High-Energy Physics: Characteristics and Impact on Resource Management, 15th IEEE International Symposium on High Performance Distributed Computing (HPDC), June 2006.

[16] T. Oreily. REST vs. SOAP at Amazon. http://www.oreillynet.com/pub/wlg/3005?wlg=yes

[17] Paxson, V. 1996. End-to-end routing behavior in the Internet, Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, vol 26.4, ACM SiGCOMM Computer Communication Review, pages 25-38, ACM Press, New York, August 1996.

[18] Garfinkel, S. 2007. An Evaluation of Amazon's Grid Computing Services: EC2, S3 and SQS, Technical Report TR-08-07, School for Engineering and Applied Sciences, Harvard University, MA. July 2007.

[19] TeraGrid. http://www.teragrid.org

[20] Odlyzko, A. M. 2000. The Internet and other networks: Utilization rates and their implications, Information Economics & Policy, vol. 12, 2000.

[21] Odlyzko, A. M. 1999. The current state and likely evolution of the Internet, in Proceedings of Globecom'99, pp. 1869-1875, IEEE, 1999.

[22] Western Canada Research Grid. http://www.westgrid.ca

[23] Ripeanu, M. and Iamnitchi, A. 2007. S4: A Simple Storage Service for Sciences, 16th IEEE International Symposium on High Performance Distributed Computing (HPDC), Monterey Bay, CA, June 2007.

[24] Palankar, M., Onibokun, A., Iamnitchi, A., and Ripeanu, M. 2007. Amazon S3 for Science Grids: a Viable Solution?, poster, 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI'07), April 2007.