Flexible, Adaptive Personal Cloud Storage

Malte Schwarzkopf
University of Cambridge Computer Laboratory
malte.schwarzkopf@cl.cam.ac.uk

I. Background

Without doubt, cloud computing is one of the current dominant trends in computing. More and more of our computation is moving into the cloud, owed to the advent of services such as Amazon's EC2. Increasingly, there are services and applications with the ultimate goal of moving the entirety of a user's virtual existence into the cloud, such as eyeOS[1] or Google's upcoming Chrome OS[2]. However, this requires cloud-based *data storage* (as opposed to *computation*), to be on par with the convenience and performance of local storage systems.

With "classic" means of local data storage, there are implicit trade-offs being made between availability and reliability of storage as we attribute different properties to different storage media.

There are a number of "cloud storage" services and backends already, most notably Amazon's S3, but also including Rackspace Cloud Files[3] and Microsoft's SkyDrive[4]. Currently, the use cases of these services, and their resale variants, are mostly *synchronisation* and *backups*, but not full-scale storage of personal data.

If, however, a truly universal data storage facility with properties analogous to those of local storage systems is considered – particularly with respect to availability, reliability and consistency guarantees – then it becomes evident that these existing systems do not provide currently the necessary and expected guarantees for read-write data access in a diverse set of use cases. According to Brewer's conjecture [4], no storage service can ever simultaneously achieve all three properties of *consistency*, *availability* and *partition tolerance*, meaning that any solution will always be a tradeoff.

II. Research Objectives

My proposed research aims to combine and refine the existing concepts of cloud computing and distributed storage systems by introducing a notion of *parametrised data storage* in which data is classified according to the guarantees required. The classification can be automatic or explicitly initiated by the user or an application. It could then be used by a storage system to provide *dynamic tradeoffs* in the orthogonal space of availability, consistency and partition tolerance.

In the end, I hope to have achieved the following:

- Identified common patterns and scenarios in personal data storage.
- Analysed existing cloud-based storage systems and their performance as well as the tradeoffs made; highlighted shortcomings and problems.
- Developed appropriate storage models that allow for optimisation through dynamic tradeoffs.
- Built a prototype system or component for an existing system that implements these and performed comparative evaluation against existing systems, either in the cloud storage space or in the distributed systems and databases space.

The potential benefits of this research are manifold: increased **user control over storage parameters** may facilitate greater trust into cloud storage, even if data is spread across organisational and geographical boundaries. Overall, an **increased utility** of online storage could be reached, resulting in a more adaptive and performant service. **Commercial viability** is given for both customer and storage provider: a distributed storage system that implements a dynamic scheduling can present significant economic advantages over existing systems. For example, large amounts of non-delay-critical data could be served from data centres in geographical areas with cheap electricity or particular bandwidth characteristics at a given time of day.

III. Proposed Work

In accordance with the above research objectives, I devised a programme of work intended to cover three years of full-time PhD research. Due to the nature of the undertaking, the later phases of the programme are still only vaguely defined, while some of the early work has already been completed in parts.

A. Work Completed

I have performed a survey of the literature on this topic and am in the process of familiarising myself with existing work and the details of various existing approaches. At the same time, I have performed some benchmarks on existing cloud storage systems, which turn out to exhibit a great deal of variance in

[1] http://www.eyeos.org
[2] http://www.chromium.org/chromium-os
[3] http://www.rackspacecloud.com/
[4] http://skydrive.live.com/

performance both between different systems and services (*between-service variance*) and between different requests to the same service (*within-service variance*). I believe these to be caused by a combination of optimisations (by the means of static tradeoffs) for particular workloads and contention on shared infrastructure.

B. The Nimbus Parasitic Storage System

In order to verify the assertions made above and to explore the design space, I have designed a simple *parasitic* storage architecture called *Nimbus*. The idea here is that Nimbus "parasitically" sits on top of different existing, storage services, creating a transparent layer through which data is distributed and replicated. The parasitism is in the fact that *Nimbus* is storing binary data chunks, rather than strictly defined types of data. A set of service-specific adapters, called *storage bugs* implement the respective API and, if necessary, convert or wrap data to fit with the assumptions the particular backend is making (see Figure 1). Inspired by the Mnemosyne steganographic storage system [5], data is distributed using the information dispersal algorithm in order to give additional resilience as well as useful privacy properties.

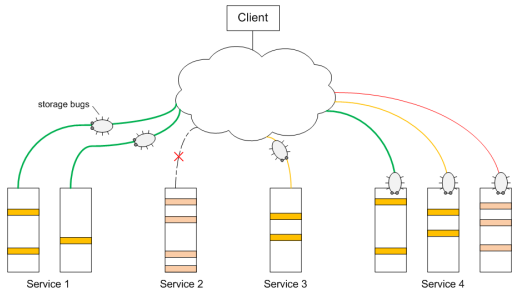

Figure 1: Schematic of the Nimbus service architecture.

The rationale for developing this concept is that a system spanning the range of existing solutions, optimised towards different workloads, may be an interesting first approach to providing high-availability personal data storage on the cloud while still mainining desirable privacy properties.

I have implemented a basic prototype in Python, and expect future work on Nimbus to primarily focus on the extension implementation of the prototype as well as a further exploration of the design space. I hope to perform testing in a real-world environment in order to ascertain what striping configuration can actually provide the bounded availability properties that I am hoping for.

Ultimately, I do not expect Nimbus to be the final answer, but I believe that the concept may have the potential of providing useful insights for an eventual open standard for global cloud storage.

C. Future Work

With Nimbus, the storage model adopted is a fairly simplistic one of storing binary "chunks" of data in a flat hierarchy and a separation of meta data from content. In the near future, I am planning to look at the general storage model and evaluate a number of potential alternative approaches, such as distributed databases.

In the longer term future, I am hoping to develop a lightweight alternative to the Nimbus approach that supports dynamic tradeoffs and encourages them through performance benefits as well as economic incentives. Ideally, this system would be incrementally deployable on existing infrastructure and merely support a new access protocol.

IV. Related Work

There are a number of existing systems that are used for wide-area data storage in cloud computing environments, though with extremely diverse use cases and optimisation decisions.

The OceanStore project [6] proposes an architecture for a complete internet-scale distributed storage system. It aims to provide a perfectly general, reliable, high performance, consistent and self-maintaining cloud storage network for data objects. The system tolerates multiple parallel failures [3], and in an approach not unlike the one taken with the IDA in *Nimbus*, erasure codes allow for blocks to be reassembled from any sufficient subset of fragments. Block-level caching is used for availability, and, again only an aggregate of nodes is seen as trustworthy.

With the Eternity service [1], Anderson presents a distributed storage model designed for maximum resilience and anonymity. This incorporates user control over parameters such as storage duration and degree of replication, and scheduling for maximal reliability to avoid DoS under all possible threat models.

A recent development is the SCADS project at Berkeley [2], built using the open-source Cassandra platform [7] that was developed by Facebook. SCADS explores models beyond simple key-value storage, and combines them with scalable, declarative consistency-performance specifications backed by machine learning mechanisms that allow adaption of the storage platform. Intrinsically geared towards providing scalable database storage for web applications, SCADS is however targetting a different storage model and also gives less consideration to privacy aspects.

References

[1] Anderson, R. J. The Eternity service. In *In Proceedings of Pragocrypt* (1996), pp. 242–252.

[2] Armbrust, M., Fox, A., Patterson, D. A., Lanham, N., Trushkowsky, B., Trutna, J., and Oh, H. SCADS: Scale-Independent Storage for Social Computing Applications. In *CIDR* (2009), www.crdrdb.org.

[3] Geels, D. M. Data Replication in OceanStore. Tech. rep., University of California at Berkeley, Berkeley, CA, USA, 2002.

[4] Gilbert, S., and Lynch, N. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *Sigact News 33* (2002), 51–59.

[5] Hand, S., and Roscoe, T. Mnemosyne: Peer-to-peer steganographic storage. In *Proceedings of IPTPS 2002* (2002), vol. 56, pp. 1–6.

[6] Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Wells, C., and Zhao, B. OceanStore: an architecture for global-scale persistent storage. *SIGARCH Comput. Archit. News 28*, 5 (2000), 190–201.

[7] Lakshman, A., and Malik, P. Cassandra: A Structured Storage System on a P2P Network. `http://code.google.com/p/the-cassandra-project/`.