The future of personal storage: online, distributed, safe and private

This article introduces Tamias, a personal storage system that leverages a cloud-like infrastructure for personal storage while retaining privacy. Tamias is an open-source framework that builds upon a strong identity infrastructure to enforce privacy during sharing and prevent personal data hijacking.

3.1 Introduction

The move to electronic storage is becoming widespread and obvious. As consumers started to embrace digital photography in the past decades, soon followed by music collections and personal videos, the amount of personal digital data has increased dramatically.

This large amount of personal digital data needs to be stored safely. However, the need for a backup solution is usually felt better by people immediately after a disaster has struck them. People resort to archiving on their personal hard disks, until they fail, within a few couple of years.

For this reason, the online storage solutions, quickly labelled cloud storage solutions, have become very popular with the personal user. An additional incentive to use cloud services is that most of those solutions also offer easy sharing with third parties. It is seen as a solution with no vulnerability and little cost. The market has become quite mature, there are many players in the industry and prices are cheap.

But these solutions have two important weaknesses in our opinion: single-provider dependence and lack of privacy. Depending on a single provider can lead to various shortcomings where a data loss occurs because of a failure from the entity or a termination of the service. The privacy aspect on the other hand is directly linked to cost. Most of the free or low-cost offers for storage depend on online advertisement as a stream of revenue. As such, they use data-mining techniques on the user's private data to build targeted profiles for advertisers. We would like to make our readers aware that there is little difference between a robot browsing their data and an actual human.

Now that we made the case for a personal storage that would be both private and distributed, we will introduce the Tamias architecture, that, in addition to the usual features of online storage, also allows private and fine-grained sharing. We will then compare it to a free storage solution, namely Dropbox*1.

3.2 The Tamias architecture

The Tamias system is a distributed storage system that solves privacy issues by using two types of encryption. In addition, storage servers can be hosted by third parties without compromising data thanks to the encrypted nature of the data. Eventually, it protects against failures thanks to it's distributed architecture and the use of erasure coding*2.

3.2.1 Encryption

The first kind of encryption used is a symmetric algorithm to protect against unwanted parsing by storage hosts. This is implemented by the Tahoe-LAFS*3 software component. In this specific component, access control is built in a distributed fashion using capabilities. They are self-certified objects that provide enough information to locate and decrypt the contents of a stored object.

As far as capability-based storage solutions are concerned, there is no strong sense of identity in the system. Usually, owning an object is actually equivalent to knowing it's capability. This piece of information can then be passed on, from user

*1 Dropbox. https://www.dropbox.com.

*2 Hakim Weatherspoon and John D. Kubiatowicz. Erasure Coding vs. Replication: A Quantitative Comparison. In First International Workshop on Peer-to-Peer Systems, 2002.

*3 The Tahoe-LAFS Foundation. Tahoe-LAFS: The Least Authority File System. http://tahoe-lafs.org.

to user, effectively spreading unbeknownst to the original author, and out of his control. In order to be able to build a fine-grained sharing mechanism that can prevent this uncontrolled spreading, we choose to associate an identity with each user.

This identity allows us to establish ownership and control capability dissemination, thus ultimately the scope of sharing.
As we want to avoid building single points of failure, we try to keep the Tamias architecture as distributed as possible. For this reason, the identity scheme is based on public-key cryptography. This is the second kind of encryption. It allows people to introduce themselves using a public key, and build a network of acquaintances through public key exchanges. Also, the lack of a central authorization server avoids the creation of another likely single point of failure.

3.2.2 Identity in Tamias

As we just described, identity is the core value of the Tamias architecture. In the case of Tamias, it takes the form of two keypairs of a public-key cryptography system. The first keypair is used to locate the user's content and is to stay secret. It is called the root keypair. The second keypair is the user identifier and allows to perform all regular public-key cryptography computations. It is called the identity keypair. The private key from this keypair is to remain secret and to be used for signature and decryption. On the other hand, the public key is to be shared with other users and to be used for encryption and signature validation.

Upon its first connection, the Tamias client offers the user to define his identity keypair. This can be done by generating a keypair (current defaults are RSA algorithm with a key length of 2048 bits) or importing an existing keypair. The root keypair is then generated and used to create a Tamias root folder in a deterministic fashion.
For identity dissemination purposes, users will then circulate the public-key of the identity keypair. On the other hand, the private key of the root keypair allows the user to configure another device using the same identity. In doing so, he obtains a consistent view of his storage space from multiple devices.

3.2.3 Fine-Grained Sharing

As explained in section 2.1, knowing the capability for an object in the storage is enough to gain access. To make fine-grained sharing possible, we must make sure that only those who received the capability legitimately can make use of it. Since it is impossible to prevent information from flowing, leaking, or being stolen, we choose to protect the capabilities. This is done by creating an authorization object. This object contains information about the actual capability, the sender identity, and the intended recipient identity. This authorization is then signed by the owner. Subsequently, we extend the storage servers so that they keep the public-key information of the owner within the stored objects. It is used by the servers to grant access to protected content. Now that the public-key is available on the client side (within the authorization) and on the server side (next to the stored object) at the same time, we manage to avoid the need of a public-key distribution infrastructure.

Per usual public-key cryptography properties, this signed authorization is impossible to hijack without knowledge of either the destination's private key or the owner's private key, thus maintaining privacy of the stored blocks. The reason is that, whenever a server is asked for a block, it will challenge the requester to prove that his identity matches the target identity recorded in the signed authorization. Of course, the server also compares the key used for signing the authorization with the key used to create the object in first place.

3.3 A performance primer

As was just discussed, the Tamias system provides some unique features that are not found in other online storage systems and are likely to incur performance penalties when comparing against less fully featured systems.

We decided to compare the file upload performance with that of the Dropbox system. It is an online personal storage system built on top of Amazon Web Services, providing a small free online storage space (or a larger storage space for a fee).
Our dataset comprises 100 pictures shot with a DSLR camera that illustrates a typical use case for private online storage. The distribution of file sizes is shown in Figure 1a.

3.3.1 Dropbox benchmarking

Using the Dropbox API, we scripted the upload and deletion of the 100 files. In order to circumvent any deduplication process that might happen on the server side, we scrambled the contents of the actual files using AES with different random keys for each run.

The results can be seen in Figure 1b. Being built on Amazon Web Service, and especially S3 storage*4, Dropbox delivers no more performance than could be obtained from S3, in the location where the test is conducted. In our laboratory environment, the time to complete a single upload to the Dropbox system is increasing linearly with the size of the file and can be summarized, with a 1.5% error rate as:

$$Time_{seconds} = Size_{(MB)} * 0.551 + 3.898$$

For these calculations, we used the 25th percentile as it gives the smallest error during fitting. Also, we didn't plot results above the 90th percentile because some transfers, for some runs, resulted in very large upload times. These large variations are caused by the fact that the traffic has to flow through the Internet towards the S3 servers, and are thus subject to interferences based on the amount of global traffic and its own variations.

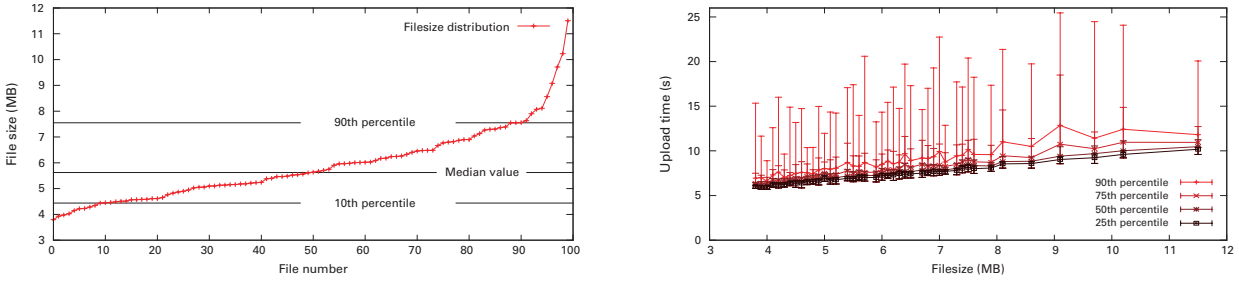
3.3.2 Tamias benchmarking

We then conducted tests of the Tamias system using an experimental deployment. The first iteration of the tests was done in optimal conditions with all nodes sharing the same LAN segment, and a total of 18 nodes. The second iteration of the tests was conducted after the nodes were shipped worldwide. We used the 14 storage nodes (Japan [7], U.S.A. [4], Sweden [3]) that were online at the time of the experiment.

Figure 2a shows the performance of the Tamias system in a stress-testing configuration, while Figure 2b summarizes the results obtained for a widely distributed testbed.

In a way, these two figures show us the best and worst case for Tamias performance. Indeed, the stress-test environment has all nodes on the same LAN, giving the best possible throughput between the client and the storage nodes. This results in a linear behavior describing the time required to complete an upload, and that can be summarized, with a 1.5% error rate, as follows:

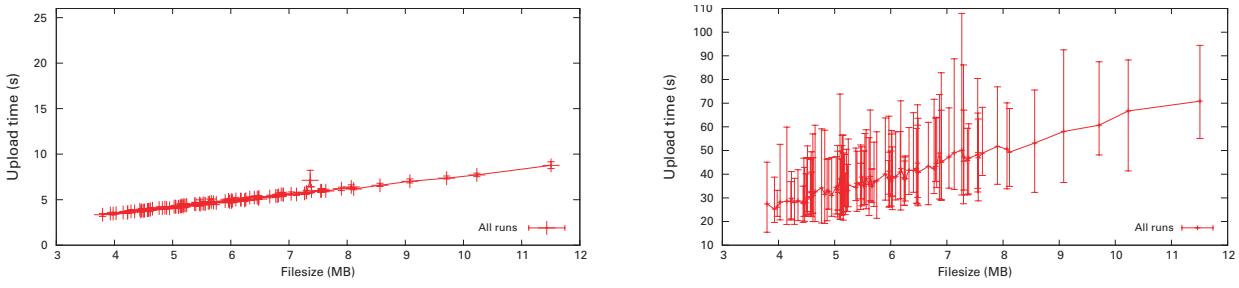
$$Time_{seconds} = Size_{(MB)} * 0.702 + 0.682$$



(a) Dataset used for experiment

(b) Dropbox upload performance

Figure 1: Dropbox benchmarking parameters and results



(a) Local deployment

(b) Worldwide deployment

Figure 2: Tamias upload time

*4 Amazon, Amazon Simple Storage Service (http://aws.amazon.com/s3/).

This result might seem disappointing with respect to the Dropbox equation, because the linear factor is bigger in the Tamias case. However, there is a large overhead on data upload in the Tamias context because data is encrypted and erasure-coded before it is sent to the various independent storage servers.

The default erasure coding settings incur a 3 times network bandwidth overhead, but guarantees that no content can be lost as soon as the transfer itself is finished. Anyway, the setup time required for the Dropbox transfer (the fixed part of the linear equation) is big enough that transfer times stay lower for our whole dataset.

As for the worldwide deployment, most nodes are located in home environments behind a NAT router. The size of the testbed and encoding parameters used are sufficient to make sure that at least one slow node will be used during the upload. Since all chunks are uploaded in parallel, the slowest node is the limiting factor. The upload takes as much time to complete as the time to complete the transfer of a chunk to this slowest node. In this worldwide setup, the linear function that fits the 25th percentile of the upload time, with 1.8% error-rate, is as follows:

$$Time_{seconds} = Size_{(MB)} * 4.903 + 2.114$$

The fixed part of the delay is still lower than the one necessary for Dropbox, but the bigger size factor makes the Tamias Worldwide deployment much less efficient for all the file sizes that were in our sample dataset.

3.3.3 Throughput considerations

Using these three equations and the file size distribution, we can compute the average time required to complete the transfer of the whole dataset, and deduce the effective throughput (i.e. the throughput that is experienced by the user for this specific dataset). These results are summarized in Table 1.

Table 1: Comparison of Effective Throughput

	Dropbox	Tamias (local)	Tamias (worldwide)
Upload Time (s)	715	482	3105
Throughput (KB/s)	845	1252	195

As far as Tamias performance is concerned, the effective throughput directly depends on the placement of storage servers. It is thus a matter of trading off geographical diversity for performance.

3.4 Conclusion

In this paper, we described the Tamias storage system. It is an online storage system for personal data that provides good reliability thanks to erasure coding and a distributed architecture. Also, two different kinds of encryptions (symmetric and asymmetric) provide both security and privacy.

We have then shown that the performance of the Tamias system obtained in best conditions is slightly better than that of the Dropbox system for all the files of our dataset. However, when the storage nodes are distributed on a worldwide scale, the performance decreases as expected. By carefully choosing the placement of the storage nodes, users can balance price (by using self-hosted nodes at home, or colocated nodes in a datacenter facility), performance (by choosing nodes with better bandwidth) and reliability (by increasing geographical diversity). This leads us to conclude that the cost for the added benefits of the Tamias storage system are affordable, because the system can be free when using nodes hosted at users' homes, while keeping performance level under control.

Authors:

Jean Lorchat
Jean Lorchat is a researcher at IIJ Innovation Institute. His research interests include wireless networking, mobile adhoc networks, layer 2 and layer 3 mobility, and more recently secure distributed storage systems. Jean got his PhD in Computer Science from Universite de Strasbourg, France in 2005 and moved to Keio University as an assistant professor until 2008 when he joined IIJ.

Cristel Pelsser
Cristel Pelsser is a researcher at IIJ Innovation Institute. Her current research interests are in Internet routing and privacy, identity in distributed storage solutions. She obtained her PhD in applied sciences from the UCL in Belgium in 2006. From 2007 to 2009, she held a post-doctorate position at NTT Network Service Systems Laboratories in Japan.