IEEE COMSOC MMTC E-Letter

Assured Deletion of Digital Files on Cloud

Patrick P. C. Lee (IEEE member), The Chinese University of Hong Kong, Hong Kong
pclee@cse.cuhk.edu.hk

1. Introduction

Cloud storage (e.g., Amazon S3 [1]) offers an abstraction of infinite storage space for clients to outsource data storage in a pay-as-you-go manner. For example, SmugMug [6], a photo sharing website, chose to host terabytes of photos on Amazon S3 in 2006. Thus, instead of self-maintaining data centers, enterprises can now outsource the storage of a bulk amount of digitized content (e.g., audio/video files) to third-party cloud storage providers so as to save the financial overhead of data management.

However, there are critical security concerns of how to protect the access of outsourced data on the cloud. We must ensure that only authorized parties can access the outsourced data on the cloud, and prohibit third-party cloud storage providers from mining any sensitive information of the client data for their marketing purpose. In particular, it is important to guarantee *assured deletion* of data, meaning that outsourced data will become permanently inaccessible to anybody upon requests of deletion. Assured deletion of data is desirable, as the data may be unexpectedly disclosed in the future due to malicious attack or careless management of cloud providers. For example, a company has archived millions of email messages among its employees and customers on the cloud, and later decides to delete them to avoid leakage of sensitive data. However, the challenge of assured deletion is that cloud providers may create and distribute multiple backup copies of data over the cloud network for reliability reasons. It is unclear whether cloud providers can reliably remove all backup copies upon deletion.

The security concerns motivate us, as cloud clients, to develop a secure cloud storage system that provides file assured deletion. However, a key challenge of building such a system is that cloud storage infrastructures are externally owned and managed by third-party cloud providers, and hence the system should never assume any structural changes (in protocol or hardware levels) in cloud infrastructures. Thus, it is important to design a secure *overlay* cloud storage system that can work seamlessly atop existing cloud storage services.

There have been practical secure storage systems (e.g., [2, 4]) that work with today's clouds, but they do not address the assured deletion of files. File assured deletion is first introduced in [5], and this idea is later prototyped atop peer-to-peer networks in [3]. However, both [3, 5] target only the assured deletion upon time expiration (which we call *time-based deletion*). They do not consider how to generalize the concept of assured deletion for different *file access policies*. For example, a file access policy may specify a set of authorized users who can access a file, and this file must be assuredly deleted if the set of authorized users changes.

2. FADE

In [7], we design and implement *FADE*, a secure overlay cloud storage system that ensures file assured deletion, while working seamlessly atop today's cloud storage services. FADE decouples the management of encrypted data and encryption keys, such that encrypted data remains on third-party cloud storage providers, while encryption keys are independently maintained by a standalone service called the *key manager*. FADE generalizes time-based deletion [3, 5] (i.e., files are assuredly deleted upon time expiration) into a more fine-grained approach called *policy-based deletion*, in which files are associated with more flexible file access policies (e.g., time expiration, read/write permissions of authorized users) and are assuredly deleted when the associated file access policies are revoked.

We now elaborate the mechanism of FADE. We associate each file with a single atomic file access policy (or policy for short), or more generally, a Boolean combination of atomic policies. Each (atomic) policy is associated with a *control key*, and all the control keys are maintained by the key manager. The file content will then be encrypted with a *data key*, and the data key will further be encrypted with the control keys corresponding to the policy combination. When a policy is revoked, the corresponding control key will be removed from the key manager. This implies that the data key and hence the encrypted content of the file cannot be recovered with the control keys of the revoked policy. In this case, we say the file is deleted, even though its physical copy (which is encrypted) may still exist, as the file is no longer accessible to anybody.

The key manager can be deployed as a minimally trusted third-party service (e.g., a server maintained by the system administrators of a company). By minimally trusted, we mean that the key manager reliably removes the control keys of revoked policies. It is possible that the key manager can be compromised. In this case, an attacker can recover the files that are associated with existing active policies. On the other hand, files that are associated with revoked policies still remain inaccessible, as the control keys are removed. We emphasize that the deployment of the key manager is independent of the cloud. Thus, we do not require any re-engineering of the cloud to support FADE.

In [7], we describe how to associate the control keys of different file access policies with individual files, and how to operate on the data and control keys so as to achieve policy-based file assured deletion. We implement a prototype of FADE that works atop Amazon S3 and empirically evaluate the performance overhead of FADE. We justify that FADE can be feasibly deployed with today's cloud storage services. We refer readers to [7] for details.

3. Conclusions and Future Work

FADE can be viewed as a value-added security service that further enhances the security properties of the existing cloud storage services. Its design goal is to provide policy-based file assured deletion via various cryptographic key operations, while working seamlessly with today's cloud storage services. Given that FADE relies on the key manager to achieve assured deletion, our future work is to improve the robustness of FADE by using a *quorum* of key managers to avoid the single-point-of-failure problem in key management.

The source code for FADE is available for download at: **http://ansrlab.cse.cuhk.edu.hk/software/fade**

Acknowledgment
This research was supported by project #MMT-p1-10 of the Shun Hing Institute of Advanced Engineering, The Chinese University of Hong Kong.

References

[1] Amazon Simple Storage Service (Amazon S3), http://aws.amazon.com/s3/.

[2] Dropbox, http://www.dropbox.com.

[3] R. Geambasu, T. Kohno, A. A. Levy, H. M. Levy, "Vanish: Increasing Data Privacy with Self-Destructing Data", *USENIX Security Symposium*, 2009.

[4] JungleDisk, http://www.jungledisk.com

[5] R. Perlman, "File System Design with Assured Delete", *Proc. of NDSS*, 2007.

[6] SmugMug, http://www.smugmug.com

[7] Y. Tang, P. P. C. Lee, J. C. S. Lui, R. Perlman. "FADE: Secure Overlay Cloud Storage with File Assured Deletion", *Proc. of SecureComm*, 2010.