Analysis of Cloud access security on file system using secure policies

Priyanka Khandelwal
M.Tech student(csc)
R.C.E.W,Jaipur,Rajasthan,India
Priyankakhandelwal85@yahoo.co.in

***Abstract*-Now a days we can outsource data backups off-site to third-party cloud storage services by which we can reduce data management costs. However, we need to provide security guarantees for the outsourced data, maintained by third parties. In this paper we design and implement FADE, a secure overlay cloud storage system which is able to achieve fine-grained, policy-based access control and file assured deletion. It associates the outsourced files with file access policies, and assuredly deletes files to make them unrecoverable by anyone upon revocations of file access policies. For achieving such security goals, FADE is built upon a set of cryptographic key operations that are self-maintained by a quorum of key managers that are independent of third-party clouds. Particularly, FADE acts as an overlay system which works seamlessly atop today's cloud storage services. In this paper we study a proof-of-concept prototype of FADE , We conduct extensive empirical studies, and demonstrate that FADE provides security protection for outsourced data, while introducing only minimal performance and monetary cost overhead. Our work provides insights of how to incorporate value-added security features into today's cloud storage services.**

Keywords:Accesscontrol,Assured deletion,backup/recovery,cloud storage

I. Introduction

Cloud computing is defined as the delivery of computing services over the internet. Cloud services allow individuals and businesses to use software and hardware services that are managed by third parties at remote locations. Examples of cloud services include online file storage, social networking sites, webmail, and online business applications. The cloud computing model allows access to information and computer resources from anywhere ,where network connection is available. It provides a shared pool of resources, including data storage space, networks, computer processing power, and specialized corporate and user applications.The cloud computing model allows access to information and computer resources from anywhere ,where network connection is available. It provides a shared pool of resources, including data storage space, networks, computer processing power, and specialized corporate and user applications.

Now a days *Cloud storage* become a new business solution for remote backup outsourcing, as it offers an abstraction of infinite storage space for clients to host data backups in a pay-as- you-go manner. It helps enterprises and government agencies significantly in reducing their financial overhead of data management, now they can archive their data backups remotely to third-party cloud storage providers rather than maintain data centers on their own. For example, A SmugMug , a photo sharing website, chose to host terabytes of photos on Amazon S3 in 2006 and saved thousands of dollars on maintaining storage devices[10] . Not only enterprises and government agencies, an individuals can also archive their personal data to the cloud using tools like Dropbox . particularly, with the origin of smartphones, we expect that more people will going to use Dropbox-like tools to move audio/video files from their smartphones to the cloud because Smart phones have limited storage space.

As we now outsource the storage of sensitive data to third parities,so security concerns become necessary. In this paper, we are particularly interested to discuss two security issues. First, we need to provide guaran- tees of *access control*, in which we must ensure that only authorized parties can access the outsourced data on the cloud. In particular, we must restrict third-party cloud storage providers from burrowing any sensitive information of their clients' data for their own marketing purposes. Second, it is important to provide guarantees of *assured deletion*, meaning that outsourced data is permanently inaccessible to anybody (including the data owner) upon requests of deletion of data. Keeping data permanently is undesirable, as data may be unexpectedly opened in future due to malicious attacks on the cloud or because of careless management of cloud operators. The challenge of achieving assured deletion is that we have to trust cloud storage providers to actually delete data, but they may be reluctant in doing so . Also, cloud storage providers typically keep multiple backup copies of data for fault-tolerance reasons. It is uncertain, from cloud clients' perspectives, whether cloud providers faithfully remove all backup copies upon requests of deletion.

These security concerns motivate us, as cloud clients, to

have a system that can accomplish access control and assured deletion of outsourced data on the cloud *in a fine-grained manner*. As we all know, building such a system is really a difficult task and especially when it involves protocol or hardware changes in cloud storage infrastructures that are externally owned and managed by third-party cloud providers.Thus, it is necessary to design a secure *overlay* cloud storage system that can be overlaid and work seamlessly atop existing cloud storage services.

In FADE, active data files that live on the cloud are associated with a set of user-defined *file access policies* like., time expiration, read/write permissions of authorized users and these data files are accessible only to users who satisfy the file access policies[7]. The design perception of FADE is to decouple the management of encrypted data and cryptographic keys, such that encrypted data remains on third-party,cloud storage providers, while cryptographic keys are independently maintained and operated by a quorum of key managers that altogether from trustworthiness.To provide gurantees of access control and assured deletion,FADE leverages off-the-shelf cryptographic schemes including threshold secret sharing and attribute based encryption[6] and performs various cryptographic key operations that provide security protection for basic file upload/download operations. We implement a proof- of-concept prototype of FADE to validate its feasibility, and export a set of library APIs that can be used, as a value-added security service, to boost the security properties of general data outsourcing applications.
The design of FADE is based on the *thin-cloud* interface which means that it only requires the cloud to support the basic data access operations such as PUT and GET. Thus, FADE is applicable for general types of storage backends, as long as such backends provide the interface for uploading / downloading data.
In short,our work address the access control and assured deletion problems from a practi- cal perspective. Our FADE implementation characterizes and evaluates the performance and monetary cost impli- cations of applying access control and assured deletion in a real-life cloud storage environment.

II. FADE Overview

We now analyze the design of FADE,A system which provides guarantees of access control and assured deletion for the outsourced data in cloud storage. In this paper we show the necessary components of FADE, and state the design and security goals that it seeks to achieve.Figure 1 illustrates an overview of the FADE system.The cloud hosts data files on behalf of a group of FADE users who want to outsource data files to the cloud based on their definitions of file access policies. It applies security protection to the outsourced data files before they are hosted on the cloud.

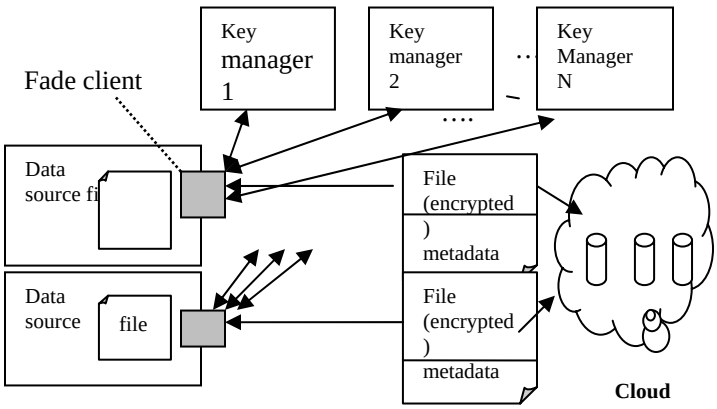

Fig. 1: The FADE system. Each client (deployed locally with its own data source) interacts with one or multiple key managers and uploads/downloads data files to/from the cloud.

2.1 FADE Entities

As shown in Figure 1, the FADE system is made up of two main entities:

- *FADE clients:* A *FADE client* is an interface that bridges the data source or filesystem and the cloud. It applies encryption/decryption to the outsourced data files uploaded to /downloaded from the cloud.

- *Key managers:* FADE is made up of a quorum of key managers,each of which is a stand-alone entity which maintains policy-based keys for access control and assured deletion.

2.2 Representation of Metadata

For every data file protected by FADE, we include the metadata. Metadata describes the policies that are associated with the file as well as a set of cryptographic keys. More precisely, the metadata contains the specification of the Boolean combination of policies, and the corresponding crypto- graphic keys including the encrypted data key of the file and the control keys associated with the policies. Here, we assume that each (atomic) policy is specified by a unique 4-byte integer identifier. To represent a Boolean combination of policies, we express it in *disjunctive canon- ical form*, i.e., the disjunction (OR) of conjunctive policies, and use the characters '∗' and '+' to denote the AND and OR operators. We upload the metadata as a separate file to the cloud. This enables us to renew policies directly on the metadata file without retrieving the entire data file from the cloud.In our implementation, individual data files have their own metadata, each specifying its own data key. To reduce the metadata overhead as compared to the data file size, we can form a tarball of multiple files under the same policy combination and have all files protected with the same data key.

2.3 Cryptographic Keys

In this paper we discuss a FADE system which defines three types of cryptographic keys to protect data files stored on the cloud:

- *Data key:* A data key is a random secret that is generated and maintained by a FADE client. It is used for encrypting or decrypting data files via symmetric-key encryption (e.g., AES).
- *Control key:* A control key is associated with a particular policy. It is represented by a public-private key pair, and the private control key is maintained by the quorum of key managers. It is used to encrypt/decrypt the data keys of the files protected with the same policy. The control key forms the basis of policy-based assured deletion.it is used for encrypting and decrypting files via AES..
- *Access key:* Similar to the control key, an access key is associated with a particular policy, and is represented by a public-private key pair. However, unlike the control key, the private access key is maintained by a FADE client that is authorized to access files of the associated policy. The access key is built on attribute-based encryption , and forms

the basis of policy-based access control. Intuitively, to successfully decrypt an encrypted file stored on the cloud, we require the correct data key, control key, and access key. Without any of these keys, it is computationally infeasible to recover an outsourced file being protected by FADE. The following explains how we manage such keys to achieve our security goals.

2.4 Security Goals

Here we specify the security goals that FADE attempt to find in order to protect the outsourced data files.

Threat model: Here, we assume an adversary that seeks to compromise the privacy of two types of files that are outsourced and stored on the cloud:

(i) *active files*,
i.e., the data files that the adversary is unauthorized to access and

(ii) *deleted files*, i.e., the data files that havebeen requested to be deleted by the authorized parties.

III. Design and Implementation of FADE

In this paper we discuss a design and working of a cloud storage system called FADE, which aims to provide access control assured deletion.FADE .

3.1 Data Owner/Client:

In FADE client implementation uses four function calls to enable end users to interact with the cloud:

3.1.1 *Upload(file, policy):* The client encrypts the input file according to the specified policy (or a Boolean combination of policies). Here, the file is encrypted using the 128-bit AES algorithm with the cipher block chaining (CBC) mode. After en- cryption, the client also appends the encrypted file size (8 bytes long) and the HMAC-SHA1 signature (20 bytes long) to the end of encrypted file for integrity checking in later downloads. It then sends the encrypted file and the metadata onto the cloud.

3.1.2 *Download(file):* The client retrieves the file and policy metadata from the cloud. It then checks the integrity of the encrypted file, and decrypts the file.

3.1.3 *Revoke(policy):*The client tells the key managers to permanently revoke the specified policy. All files associated with the policy will be assuredly deleted. If a file is associated with the conjunctive policy combination that contains the revoked policy, then it will be assuredly deleted as well.

3.1.4 *Renew(file, new_policy):* The client first fetches the metadata of the given file from the cloud. It updates the metadata with the new policy. Finally, it sends the metadata back to the cloud. Note that the operation does not involve transfer of the input file.We export the above function calls exported as library APIs. Thus, different implementations of the client can call the library APIs and have the protection offered by FADE. In our current prototype, we implement the client as a user-level program that can access files under a specified folder.

IV. Sequence diagram

An interaction diagram that shows how process operate with between mobile user/user ,key manager and cloud and in what order.

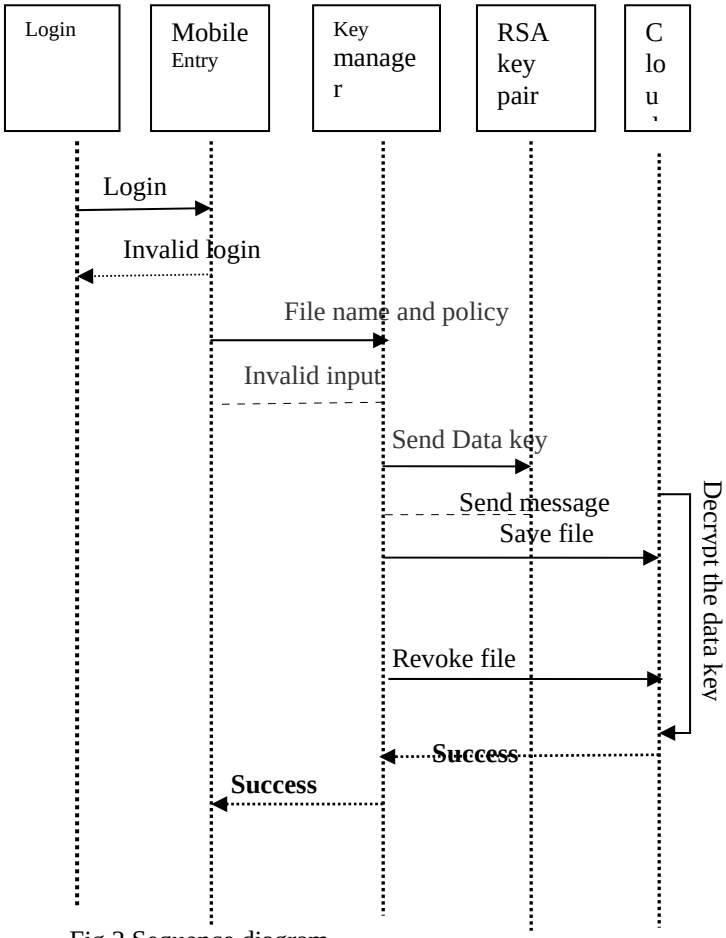

Fig 2 Sequence diagram

V. Evaluation

In this paper we study that Time-based file assured deletion, is introduced, which means that files can be securely deleted and remain permanently inaccessible after a pre-defined duration. The main idea behind it, that a file is encrypted with a *data key* by the owner of the file, and this data key is again encrypted with a *control key* by a separate key manager. The key manager is a server that is responsible for cryptographic key management. In, the control key is *time-based*, meaning is that it will be completely removed by the key manager as an expiration time is reached, the expiration time is specified at the time the file is first declared. Without
the control key, the data key and hence the data file remain encrypted and are deemed to be inaccessible.

Problems Of Existing System

1) Without the control key, the data key and hence the data file remain encrypted and are deemed to be inaccessible.

2) The main security property of file assured deletion is that even if a cloud provider does not remove expired file copies from its storage those files remain encrypted and unrecoverable.

VI. Conclusions

In this paper we study that with the origin of cloud computing the conventional way of computing has gone for a sea change. As per some expert opinions, it is going to be the face of future cloud computing. And hence, the future of cloud computing seems very promising. And this is not an overestimate or exaggeration, as we are already using cloud and its applications in one form or another. Using email and connecting to social media through smart phones, watching movies over smart phones and uploading and accessing pictures from websites like Flicker are common examples of cloud computing in our day-to-day life. However, privacy and integrity concerns become relevant as we now count on third parties to host possibly sensitive data. To protect outsourced data, a straightforward approach is to apply cryptographic encryption onto sensitive data with a set of encryption keys, yet maintaining and protecting such encryption keys will create another security issue. One specific issue is that upon requests of deletion of files, cloud storage providers may not completely remove all file copies (e.g., cloud storage providers may make multiple file backup copies and distribute them over the cloud for reliability, and clients do not know the number or even the existence of these backup copies), and eventually have the data disclosed if the encryption keys are unexpectedly obtained, either by accidents or by malicious attacks. Therefore, we seek to achieve a major security goal called file assured deletion, meaning that files are reliably deleted and remain permanently unrecoverable and inaccessible by any party.

REFRENCES

[1] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon. RACS: A Case for Cloud Storage Diversity. In Proc. of ACM SoCC, 2010.

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data.In Proc. of ACM CCS, 2006.

[3] S. Kamara and K. Lauter.Cryptographic Cloud Storage. In Proc. of Financial Cryptography: Workshop on Real-Life Cryptographic Protocols and Standardization, 2010.

[4] LibAWS++. http://aws.28msec.com/, 2010.

[5] A. J. Menezes, P. C. van Oorschot, and S. A.Vanstone. Handbook of Applied Cryptography. CRC Press, Oct 1996.

[6] W. Stallings. Cryptography and Network Security.Prentice Hall, 2006.

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

[8] C. Wang, Q. Wang, K. Ren, and W. Lou. Privacy-preserving public auditing for storage security in cloud computing. In Proc. of IEEE INFOCOM, Mar 2010.

[9] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu.Plutus: Scalable Secure File Sharing on Untrusted Storage. In Proc. of USENIX FAST, 2003.

[10] SmugMug. http://www.smugmug.com/, 2010.

.

,