Securing Data in Cloud Storage with File Assured Deletion

Anandavalli. M
Department of Computer Science,
T.S. Narayanaswami College of Arts and Science,
Navallur, Chennai - 603103. Tamilnadu, India

L. Mary Immaculate Sheela, Ph.D.
Department of Computer Science and Engineering,
R.M.D. Engineering College,
Kavaraipettai - 601206, Tamilnadu, India

ABSTRACT
Cloud Storage, Policy – based file assured deletion Cloud computing offers the hardware and software resources are made available on the internet. These services are provided and managed by the third party cloud providers at remote locations. Cloud storage providers are responsible for keeping data available and accessible which reduces the management overhead of the cloud. Cloud storage services are accessed through web content management or API. FADE (File Assured Deletion), a secure overlay cloud storage system constitutes policy-based access control and file assured deletion. It associates each file with access policies are unrecoverable by anyone. It protects the deleted data with policy-based file assured deletion. Our paper focuses on providing privacy and integrity of the outsourced data through FADE.

Keywords
Cloud Storage, Policy-based file assured deletion

1. INTRODUCTION
Cloud storage is solution for remote back-up outsourcing. It offers the infinite storage space for client to host data backups in a pay-as-you –go-manner. It helps the enterprises archive their data backups to third party cloud storage providers rather than maintaining it on their own. It reduces the financial overhead of data management. Two security issues are focused in this paper. i) Data files are protected from unauthorized access. ii) Data files are deleted only by the authorized parties. It gives the guarantee of assured deletion that means, the outsourced data is permanently inaccessible by anybody (including the data owner).

Assured deletion depends upon the trustworthiness of the cloud storage providers who actually delete data, but they may keep multiple backup copies of data. The cloud client point of view is that whether cloud providers reliably remove all backup copies upon the requests of deletion.

In this paper, we propose FADE, a fine grained access control and assured deletion for outsourced data on the cloud. FADE supports different techniques for assured deletion. They are,

- Policy-based file assured deletion
- Time-based file assured deletion
- Custom classes that can be deleted on-demand.

Our paper focuses on Policy-based file assured deletion

2. SYSTEM ANALYSIS
2.1 Existing System
In time-based file assured deletion, files can be assuredly deleted and inaccessible to anyone when it reaches the expiration time. A file is encrypted with the data key and sends the data is further encrypted with the control key by a separate key manager. Here, the control key is time-based, meaning that files will be completely removed when an expiration time is reached [1]. The expiration time is specified when the file is first created. But the only disadvantage of this technique is that we cannot recover the data key and data file without the control key when it reaches the expiration time. So, it is still encrypted and inaccessible.

2.2 Proposed System
We propose a cloud storage system with policy-based file assured deletion. In which, we associate each files with access policies that controls access privileges on it. Here the files are assuredly deleted and inaccessible by anyone when their associated policies are revoked. FADE is implemented through a set of cryptographic techniques which include ABE (Attribute Based Encryption) scheme and a quorum of key managers. The main advantage of this technique is that, the client can get all security services which are provided by the cloud.

3. SYSTEM OVERVIEW
3.1 Participants in the System
3.1.1 FADE Client
They acted as an interface between the data source or file system and the storage cloud. It may be a client application, a user-level program, file system of a PC or a mobile device [1]. At first, the data owner requests the key manager to decrypt the blinded version of the encrypted data key. Then the key manager will check the policy for decryption. If the associated policy is satisfied, then the key manager will decrypt the data key and return the blinded version of the original data key. Finally, the FADE client will recover the data key from blinded version. Hence, the content of the actual data key remains unknown to the key manager and the attacker also.

3.1.2 Key Manager
The key managers maintain the policy-based control keys that are used to encrypt the data keys. They perform the encryption, decryption, renewal, revocation processes based upon the request made by the FADE clients. File assured deletion is achieved by the key manager who reliably removes the control keys of the revoked policies. Here the files remain inaccessible because the control keys are removed.

3.1.3 Storage Cloud
It is maintained by the third party cloud provider (e.g. Amazon S3) and keeps the data on behalf of the data owner[1]. Whenever the data owner needs the data key they can access it from the storage cloud through the internet.

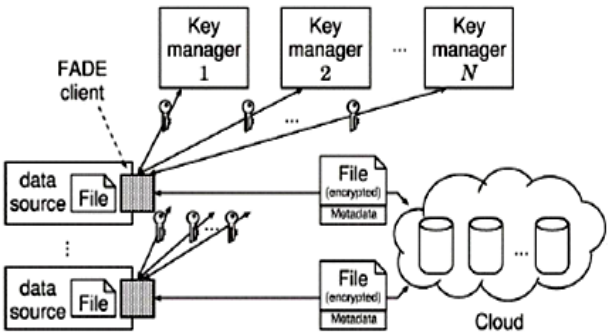

Fig 1: The FADE system

3.2 Key Management

FADE system supports three types of cryptographic keys to protect the data which is stored on the cloud

3.2.1 Data Key

A data key is generated and maintained by a FADE client. It is used to perform the encryption and decryption process on data files through symmetric key encryption.

3.2.2 Control Key

A control key is associated with each policy. It is a combination of public-private key pair and the private control key is maintained by the quorum of key managers [6]. Control key is used for performing encryption and decryption process on data keys and the files are protected with the same policy. Policy – based assured deletion is performed through the control key.

3.2.3 Access Key

An access key is associated with a particular policy and it is specified by public-private key pair. Unlike control key, the private access key is maintained by the FADE client who is authorized key to access files of the related policy. It is built on attribute based encryption and forms the basis of policy-based access control.

We require a proper control key, data key, access key to perform the decryption process of an encrypted file.

4. PRESENTATION OF METADATA

4.1 File Metadata

It contains the information about the file size and Hash function. Hash function is used to encrypt the file metadata and that will be uploaded to the storage cloud as a single file.

4.2 Policy Metadata

It specifies the Boolean combination of policies and the encrypted cryptographic keys. Boolean combination of policies are expressed in disjunctive canonical form, and use the characters '*' and '+' to denote the AND & OR operators [1]. The metadata is uploaded as a separate file to the cloud. It enables us to renew policies directly to 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. It ensures that multiple files under the same policy combination and protected with same data key. Policy metadata is specified by a unique 4-byte integer.

5. DESIGN AND IMPLEMENTATION OF FADE

5.1 Functions of the Data Owner/Client

The data owner uses the following function calls to enable the end user to interact with the storage cloud.

5.1.1 Upload (file, policy)

The data owner encrypts the input file using the specified policy. It then sends the encrypted file onto the cloud.

5.1.2 Download (file)

The data owner retrieves the file and policy from the cloud, checks the integrity of the file and decrypts the file.

5.1.3 Delete (file)

The data owner tells the key manager to permanently cancel the specified policy. All files linked with the policy will be assuredly deleted.

5.1.4 Renewal (file, new-policy)

The data owner fetches the policy for the given file from the cloud. It then uploads the old policy with new policy. Finally, it sends the policy to the cloud.

5.2 Functions of the Key Manager

5.2.1 Creating a policy

The key manager creates a new policy and returns the public control key[1].

5.2.2 Retrieving the public control key of a policy

If the policy is accessible, then the key manager returns the public control key otherwise it returns an error[1].

5.2.3 Decrypting a key with respect to a policy

If the policy is satisfied, the key manager decrypts the key. Otherwise it returns an error.

5.2.4 Revoking a policy

The key manager revokes the policy and removes the corresponding keys.

All keys are built upon 1024-bit blinded RSA with quorum of key managers

6. POLICY-BASED FILE ASSURED DELETION

In policy-based file assured deletion each file is associated with a single file access policy or Boolean combination of policies. Each policy is associated with a control key and maintained by the key manager. The file is encrypted with a data key. Then the data key is again encrypted with control keys corresponding to the policy combination. When the policy is revoked, the related control key will be removed from the key manager. Thus, policy combination associated with a file is revoked and no longer holds the data key and hence the encrypted content of the file cannot be recovered with the control key of the policies. In this way file assured deletion is achieved.

6.1 Defining Policies

Policies are defined by two ways.

6.1.1 User-based policy

Policy is based upon the user constraints. E.g.: p1: Alice is a doctor.

6.1.2 Time-based policy

Policy is based upon the time constraints. E.g.:p2: She has to come for the duty at 10.a.m.

6.2 Multiple policies

FADE supports a Boolean combination of various policies. It is classified into two logical connectives.

6.2.1 Conjunctive Policies

All policies should be satisfied when we recover the file. E.g. p1∧p2.

6.2.2 Disjunctive Policies

Any one of the policy has to satisfy when we recover the file. E.g. p1∨p2.

7. BASIC OPERATIONS OF FADE

7.1 Notations

In policy-based file assured deletion, each policy 'i', the key manager generates two secret RSA numbers pi, qi and compute $n_i = p_i.q_i$. Then the key manager randomly selects the public-private control key pair (e_i,d_i). Then, (n_i,e_i) will be publicized. d_i is securely stored in the key manager. When the data owner encrypts a file F, it randomly generates the data key K, and the secret key s_i that corresponds to policy p_i.

Table 1. Notation

S.No	Notation	Definition
1	p_i	Policy with index i
2	p_i, q_i	Prime numbers used in RSA for policy p_i
3	n_i	$n_{i=} p_i. q_i$
4	(e_i,d_i)	Public/Private control key pair for policy p_i in RSA
5	s_i	Secret key for policy p_i
6	{}Key	Symmetric key encryption with key
7	R	Random number for RSA
8	F	File generated by the client
9	K	Data key used to encrypt the file.

7.2 File Upload

The data owner requests the key manager for the public key (n_i,e_i) of policy pi. The data owner generates two random k and s_i, They send {k} s_i, $s_i^e{}_i$, and {F} k to the cloud.

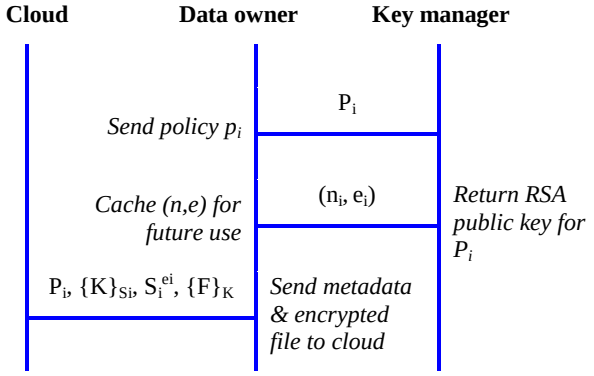

Fig 2: File Upload

7.3 File Download

The data owner gets $\{K\}s_i$, S_i^{ei}, and {F} k from the storage cloud. The data owner generates a secret random number R, computes R^{ei} and sends S_i^{ei}. $R^{ei} = (S_iR)^{ei}$ to the key manager to requests for decryption. The key manager computes and returns $((S_iR)^{ei})^{di} = S_iR$ to the data owner. The data owner can now remove R and obtain S_i decrypt $\{K\}s_i$ for the file {F}K.

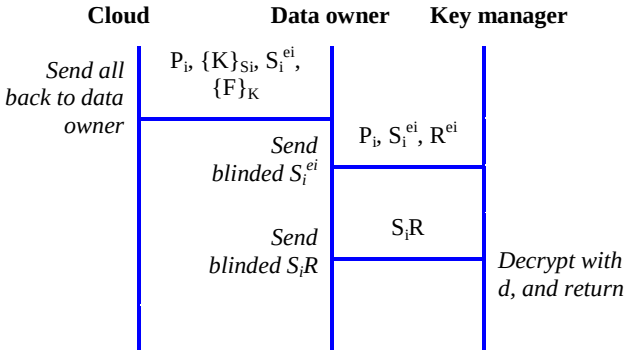

Fig 3: File Download

7.4 Policy Renewal

Policy renewal is implemented by combining the operations of file upload and download without retrieving the encrypted file from the cloud. The procedure of policy renewal is summarized as follows:

All encrypted keys are downloaded from the storage cloud.

Send it to the key manager for decryption.

Recover the data key.

Data key is re-encrypted with the control keys of the new policies.

The newly encrypted keys are sent back to the cloud.

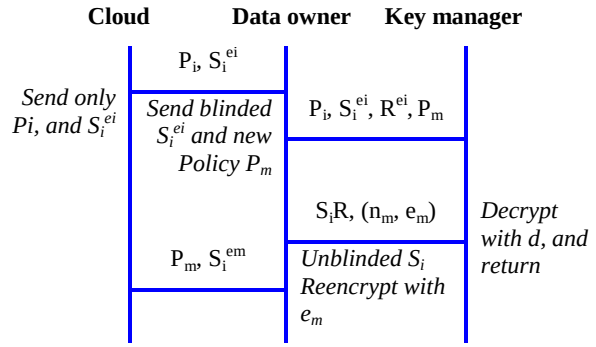

Fig 4: Policy Renewal

7.5 Time-Performance of FADE

7.5.1 File Transmission Time

It represents the uploading/downloading time for the data file between the client and the cloud[5].

7.5.2 Metadata Transmission Time

Time for uploading/downloading metadata files with policy information and the cryptographic keys to the cloud.

7.5.3 Cryptographic Operation Time

It represents the time to perform the AES and HMAC operation in the file.

8. FUTURE ENHANCEMENTS

8.1 Adding an additional layer of encryption

Data owner encrypts the file with long-term secret key and re-encrypts the encrypted file with data key. If the key manager collude the file still the data owner has the encrypted file.

8.2 Multiple files with same policy metadata

File is maintained as batch-based approach. Multiple files with same policy metadata and same set of cryptographic key reduces the storage overhead of policy metadata.

9. CONCLUSION

We propose a cloud storage system called FADE, Which goal is to provide access control and assured deletion of files with file access policies. We also present the file upload, file download operations with different types of cryptographic keys. The security goal is achieved through FADE in which, the files are reliably deleted and remain permanently unrecoverable and inaccessible by any adversary. In our future work, this scheme will be implemented into a public cloud like Amazon S3 to prove the security of this scheme. The security of the key management will be enhanced by adding a new entity called trusted authority with the quorum of key managers.

10. REFERENCES

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

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

[3] R. Perlman. 2007 File System Design with Assured Delete. In ISOC NDSS.

[4] S.Kamara and K.Lauter. 2000. Cryptographic Cloud Storage. In Proc. of Financial Cryptography: Workshop on Real-Life Cryptographic Protocols and Standardization.

[5] Priyanka Nagtilak and Archana Lomte. 2013. Dynamic Access Control and File Assured Deletion For Secured Cloud Storage. International Journal of Research in Advent Technology, Vol.1, No.5, (Dec 2013), 214-215.

[6] Priyanka Khandelwal. 2013. Analysis of Cloud access security on file system using secure policies. International Journal of Computer Science and Information Technology & Security. Vol.3, No.6, (Dec 2013), 406.

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

[8] Rutuja R. Sadul (et.al)., 2014. A Survey of Different Encryption Techniques for Secure Cloud Storage. Multidisciplinary Journal of Research in Engineering and Technology. Vol.1, No.1, (Apr 2014).

[9] Pallavi D.Patil, Ranjana R.Badre. 2014. Access Control and File Deletion as a Service in Cloud Computing. International Journal in Computer Technology and Applications. Vol.5, No.3, (May 2014), 1059.

[10] G. Gayatri, B. Sowmya. 2013. FADE: A Secure Overlay Cloud Storage System. International Journal of Science and Research. Vol.2, No.6, (Jun 2013), 150.