# A SELF-DESTRUCTION SYSTEM FOR DYNAMIC GROUP DATA SHARING IN CLOUD

**Ranjith.K[1], P.G.Kathiravan[2]**

[1]*PG Scholar, Department of Information Technology, V.S.B Engineering College, Tamilnadu, India*
[2]*Assistant Professor, Department of Information Technology, V.S.B Engineering College, Tamilnadu, India*

## Abstract

*Cloud computing, a recent computing technology which entirely changed the IT industry since it started to deliver resources as services through a single click. Even though cloud offers different services among storage as a service is widely used one, where users are free to store any information irrespective of its content. But, security and privacy becomes the issues here. Multi-owner data storing and sharing in a dynamic environment dumps huge amount of data files in the cloud, which remains in cloud for indefinite period of time. Since it remains there for a long period of time, the sensitive information stored may misused by any miscreant or even by service providers. To maintain cloud file's security and privacy regular removal of unwanted files is needed. A self-destructing system can be used to remove unwanted files automatically when the predefined time period for sharing specified by data owner has been expired. Such a system destructs the decryption keys immediately after time expiration followed by deletion of all replicas of the data file. By using RSA algorithm and Shamir's secret sharing scheme such a system can be developed.*

***Keywords:*** *cloud computing, cryptography, decryption key, dynamic group, privacy, secret sharing, service, self-destruction, security.*

----------***----------

## 1. INTRODUCTION

Cloud computing is a new generation computing paradigm where large pool of systems connected together via internet with the intention of resource sharing. Based on the type of applicability, cloud provides wide range of services such as Software as a service (Saas), Platform as a service (Paas), Infrastructure as a service (Iaas) etc. Many service providers came forward to offer cloud services in real world among Google, Amazon, Microsoft etc are the initiators. Low maintenance characters and ease of use, makes cloud computing different from other computing techniques. Since cloud and its services are applicable at each level, it reached across all categories of users. Almost all organizations try to use cloud to outsource their company related data. Also, introduction of mobile internet accelerated peoples to move towards cloud and its applications. To monitor and manage files stored in cloud, a cloud manager has been appointed, also called as cloud data owner. Cloud owner is responsible for granting access to users who wish to see the file contents. Permission is granted to such users only after successful verification of user credentials.

Earlier cloud systems were traditional single owner system, where data sharing couldn't reach to all relevant users due to reduced cloud utilization and manageability. As a solution multi-owner cloud systems introduced which offers maximum cloud utilization, improved reach ability of shared files etc. Multi-owner systems are groups dynamic in nature, into which any member can join or leave the system at any point in time. Such a group includes several group members and a single group manager. Each group members are data owner of his/her data files and free to store and share any information in the group. Group manager is responsible for management and monitoring of the entire group. As people rely more in cloud they are free to upload and share more files than usual. But security and privacy issues still remained as challenges. Moreover, shared files remains in cloud for indefinite period of time which increased possibilities of data misuse by cloud group members as well as cloud service providers. Also this system consumes more space unnecessarily because of the presence of stored and shared files in the cloud. Hence, periodical removal of shared files is the only solution to tackle such a situation.

Since cloud group consist large no of files, manual removal is not practical in real world. So, automatic removal of shared files is needed. A self-destruction system is the only option to solve such a problem which automatically removes unwanted files periodically. But, still it experience problems in removing files. Because, some files may need for a long time sharing whereas few may not. Therefore, automatic removal of files is not applicable here, as it can't differentiate which file is needed and which is not. It's clear that the entire system is based on time; the same can be applied for file storage and sharing. Each data owner has to specify the time duration up to which the files are available for sharing in the cloud while uploading the files to the cloud. Beyond which the files can be removed automatically.

To get a clear idea about the concept, let us consider a real life scenario of a university. University consist different departments which can be considered as a dynamic group. Department staffs are the members of the group. Any department staff can join or leave the group at any point in time. Professors can upload circular, internal marks of the students, practical exam details etc in the group. As days passed, old data files dumped in the cloud server which leads to security and privacy issues, storage space requirements etc. So it is difficult to remove old files manually. Hence, automatic removal is needed. In such a system, professors can upload circulars, other important memos into the cloud with time duration for self destruction.

To tackle above listed challenges, we propose a self-destruction system for dynamic group data sharing in cloud. To achieve such a system, the following contributions were made.

1. We propose a self-destruction system for dynamic group data sharing in cloud where any group member can share data files by specifying time duration up to which shared files available in the cloud.
2. Our proposed scheme supports dynamic groups that allow any group members to join or leave the group at any point in time. Self destruction system automatically removes unwanted files after user specified time period has expired.
3. Proposed scheme allows secure sharing of data files by leveraging Shamir's secret sharing scheme.
4. Encrypted secret can be decrypted by recombining individual key shares of group members satisfying minimum threshold count.

## 2. RELATED WORK

In [1], Liu et al. proposed a new scheme called Mona that supports dynamic grouping and sharing system, where any member is allowed to join or leave the group at any point in time. Group signature identifies and authenticates each member in the group to ensure the quality of the secret. User revocation performed by using a revocation list generated by cloud server. Each newly joined member can decrypt files without contacting data owners before his/her participation in the group. Revocation can be simply performed without making trouble in updating secret keys of other group members.

Kallahalla et al. [4], proposed a cryptographic storage system which enables secure sharing of files on untrusted server called Plutus. The large files are divided into no of small files each then encrypted with a secret key. Relevant users receives corresponding key to decrypt each small file blocks. Even though the system is flexible enough, heavy key distribution overhead experienced for large scale file sharing. Moreover, inorder to perform user revocation entire key pairs has to be updated.

In [2], Yu et al. brought a secure, scalable and fine grained data access technique in cloud with the use of Key Policy-Attribute based Encryption (KP-ABE) method. The data owner usually the group manager selects a key randomly to encrypt files and the chosen random key given to corresponding users for decoding the secret. Decryption only performed if and only if it satisfies predefined access structure of the file. This system also requires updating secret keys of all users to perform a user revocation operation.

Zeng et al. [9], proposed SeDas system for self destruction of stored files in cloud based on active storage framework. SeDas scheme stores files as separate nodes each keep information about time to live property value for that particular node. Once ttl reached expiration time, it automatically deletes the node from the storage system. This is an object based storage system which receives request from client to create a storage node. By using user provided information it develops a node. Metadata server is available in the system to monitor and manage entire operations in the system.

A content distribution system was proposed [10] which consist two layer architecture in which first layer responsible for distribution of data files among peers whereas the second layer for providing feedback on the distribution. The focus of this work is controlled distribution of confidential data across peers. Here the distributed content is one time use on a trusted peer site. After successful access the key automatically becomes invalid which does not support further access. If any access found to be unauthorized, the file destructs itself.

Vanish system [11] proposed timeout concept for secure data storage in cloud which encrypts secret files and splits the key into shares and later stores the shares in Distributed Hash Table (DHT). Vanish system consist encapsulation and decapsulation processes. A Vanish system encapsulates user's data file into a VDO object and uploads into cloud. This VDO objects may be transmitted or copied etc but it can't read without decapsulating parameters. The encapsulated VDO objects consist key to decrypt, fixed timeout, encrypted data and minimum no of shares needed to decrypt the secret. To decapsulate the VDO object, users need decryption key shares. If someone failed to decapsulate VDO object before fixed timeout, then it destructed forever.

Ephemerizer [13], similar to vanish system uses the same time out concept, but a little bit different. Ephemerizer system keeps separate server to store decryption keys rather than DHT and performs the same encapsulation and decapsulation operations. Ephemerizer server takes atmost care in managing and distributing key pairs across multiple locations. It sends decryption keys to user until timeout expires.

Cascade [12] system which is an improved version of vanish system that combines multiple key storage mechanism into a single system. Cascade system can be attacked only if the attacker can compromise all components of the system. It supports adding new components to the system that strengthens

the system against attacks. The same encapsulation and decapsulation is applicable here.

Self-destruction system has improvements from different angles and reaches up to centralized and decentralized approach [14]. Such a system uses a new algorithm called recursive secret sharing for secret sharing process. A third party server is available to manage key shares. Also, a public DHT used to distribute secret keys to public. Two different type of encryptions used here are data encryption and secret sharing encryption. Data encryption is to encrypt user data, whereas secret sharing encryption to encrypt secret keys.

From these points, we can observe that a self destruction system can ensure security, confidentiality and integrity in storing and sharing data files in a multi-owner dynamic group.

## 3. SYSTEM MODEL AND DESIGN GOALS

### 3.1 System Model

We consider a cloud storage system with university department as an example. The department consist department heads, Assistant professors, professors and students etc. each act as components of the system. The system model constitutes three entities: the cloud server, group admin, group members. The prescribed model is given in the Fig.1.

The cloud server is a large storage of data files that uploaded by the cloud users. Cloud service providers (CSPs) are responsible for managing and monitoring cloud in addition to providing extra services. CSP provides support for cloud users to store and retrieve shared files. We assume that cloud is untrusted under certain circumstances.

Group admin is responsible for system initialization, user registration process verification and approval, revocation of user, revealing real identity of group member when any disputes rose. Usually the group admin is the higher official of the organization or team leader of the group. Therefore, we assume that group admin is trusted by other parties.

Group members are set of registered users in the dynamic group at a given time. These users are all allowed to store and share their private/public data in the cloud server. Usually the group members are the team members or staffs in the organization or department. Group membership can be dynamically changed, because of new staff registration, resigning a job in the organization.

### 3.2 Design Goals

To achieve security and privacy for stored data files, our design guarantees the following goals.

**Access control:** Group members including group admin able to access cloud if they posses valid decryption keys. Non-members and revoked members are strictly banned from accessing the cloud and its services.

**Data Confidentiality:** Unregistered or registered users without access keys can't understand the content of stored data files including the cloud.

**Anonymity and Traceability:** Anonymity guarantees tension free access to cloud without revealing real identities of cloud users. Traceability reveals real identities of cloud users if any discrepancies found.

**Efficiency:** Any registered user can upload files into cloud with timeout for triggering self destruction of the same. After time out the decryption keys and all replicas of the file deleted permanently from the cloud and even data owner can't recover those files. However the data owners are allowed to change their time out period, if desired.

## 4. PROPOSED SCHEME

### 4.1 Overview

Cloud and its services are highly influenced by people and organization. Many of are migrating to cloud for their data or related applications. Storage service is widely deploying service of cloud. So that, cloud user feels free to store and share huge no of files. Unfortunately, most of them never think about those files after sharing. Since the shared files remains in cloud for long period of time, it raises security and privacy issues in cloud groups. The shared files may include sensitive information which may misused by any miscreant or even service providers. Another issue is that dumping of huge no of files in cloud consumes more storage space and reduces search efficiency of the system. To resolve these issues we proposed a self-destruction system that automatically removes shared files after certain time period specified by its owner. The following sessions describes design and implementation details of our scheme in detail.

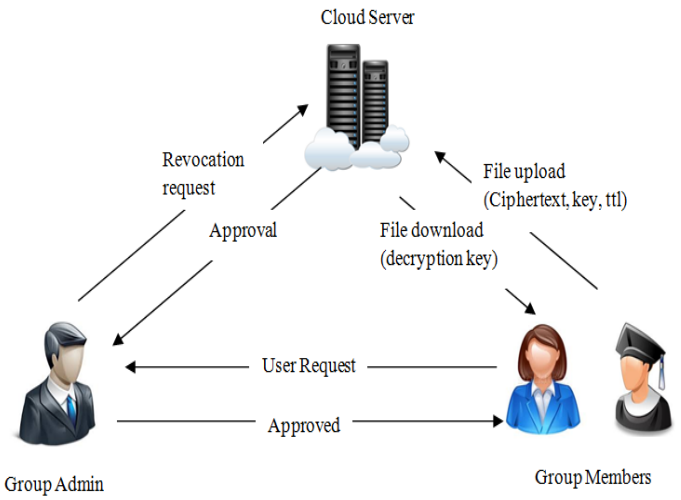


**Fig.1** System Model

### 4.2 Scheme Description

**System Startup:** System startup refers to initialization of system by Group Admin. Admin receives requests from users for registration. After successful verification of registration details registration process completed by providing user account to the candidate.

**User Registration:** A user *i* with identity $ID_i$ register with the system by providing necessary details. Group Admin verifies all the provided details and ensures his/her participation if all details are relevant to the group rules and regulations. Group admin then provides login details such as user name and pass word to the respective users.

**User Revocation:** User Revocation is the process of removal of a user from participating in the group under certain conditions. Revocation process is authorized to perform by Group Admin through a publicly available revocation list. If the login credentials of the specified user matches with the details of revocation list then access denied.

**File Upload:** File upload is the process of storing specified data files into the cloud for sharing in the group. Uploaded files remains in the cloud up to the time specified while uploading the file. Before uploading the file, file has to be encrypted by using any encryption algorithms to ensure security and privacy of the files. Here, we use Algorithm (1) for encrypting the file. Encrypted files then encapsulated with corresponding decryption key and time to live (ttl) value for the file. After successful uploading of the file, decryption key is divided into no of shares by using Algorithm (2) and each shares distributed across all the participants in the group. Decryption of the file is possible if and only if all the decryption key shares that satisfy minimum threshold count.

**Algorithm (1).** RSA Algorithm
**Input:** Two different prime numbers, integer
**Output:** Public and private key Pairs
**Begin**
**Key Generation ()**
Select p, q where p, q are prime and p! = q
Calculate n = pXq
Calculate f (n) = (p-1) (q-1)
Select integer e gcd (f (n), e) = 1; 1<e<f (n)
Calculate d $d= e^{-1}(\text{mod } f(n))$
**Return**
Public key PK= {e, n}
Private key PR= {d, n}
**End**

**Encryption()**
Plaintext m<n
Ciphertext $c=m^e \text{ mod } n$

**Decryption()**
Ciphertext c
Plaintext $m=c^d \text{ mod } n$

**Algorithm 2(a).** Shamir's Secret Share computation
**Input:** Finite field F, Secret data S ε F, min threshold k, no of shares n.
**Output:** Shares $S_1$, $S_2$, $S_3$, .... $S_n$.
Set $f_0=S$
Uniformly generate coefficients $f_1, f_2, ..., f_{k-1} \varepsilon F$
Construct polynomial $f(X)=f_0+f_1X+.....+f_{k-1}X^{k-1}$
Evaluate the polynomial $S_i=f(i)$, (i=1, 2, ...n)

**Algorithm 2(b).** Shamir's Secret Share reconstruction
**Input:** Finite field F, Shares $S_1$, $S_2$, $S_3$, .... $S_n \varepsilon F$
**Output:** Secret data S.
Compute reconstruction coefficients bi
Compute $f(0)=S_1b+S_2b+....+S_kb$
Return S=f(0)

**File Deletion:** Since the system is a self-destruction system, explicit deletion mechanisms not required. The system itself automatically removes the shared files by the time specified during upload process.

**File Access:** To access the files stored in the cloud, group members need to combine all the key shares that satisfy (*K*, *n*) threshold secret sharing scheme. Where, *K* represents the key from *n* users and *n* represents minimum number of shares needed to decrypt the file. Use revocation list obtained from the cloud to check validity of *n* users to be participated in decryption process.

**Traceability:** Traceability allows group admin to get personal identity of users in case of any discrepancies in accessing and storage. Inorder to trace a particular user, his/her cloud group identity or group signature is more than enough. So users are not supposed to share unwanted files in the group.

## 5. SIMULATION AND RESULTS

The proposed system has been simulated by using Microsoft visual studio asp pages as client interfaces and SQL server as cloud server. Cloud system is initialized by Group admin who is responsible for overall management of the system. Group members send request for registration to the admin through client interfaces developed in asp pages. All the requests registered in cloud server developed in SQL server. Group admin login to the system and verifies all registration details.

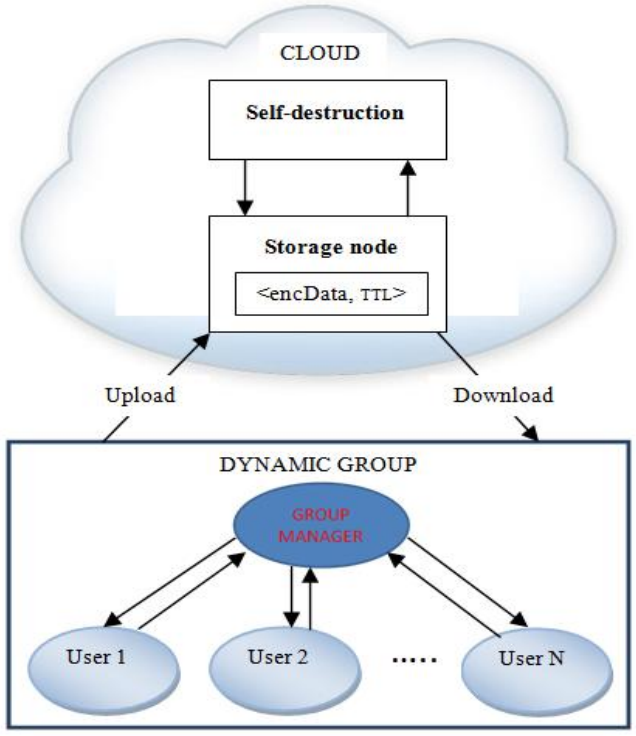


**Fig.2** System Architecture

If all information provided matches with group requirements registration request has accepted and that user can start to use cloud services. While uploading files itself the user can specify time duration up to which files may be allowed for sharing. After which files can't be viewed by even data owners.

From above simulation, the following results are obtained: uploading and managing files in cloud has less complexity since all files removed automatically after user defined time period has just expired.

## 6. CONCLUSIONS AND FUTURE WORKS

In this paper, we proposed a self-destruction system for dynamic group data sharing in cloud systems. Since shared data items in dynamic groups remains long time in the system will considerably reduce the security and privacy of system with increased complexity in managing data files. Hence, in this self-destruction system all files are removed automatically if those are no more needed. Also, the time period for sharing can be explicitly fixed by data owners while uploading the files itself. We strongly believe that the system will reduce complexities in managing old data files and thereby increasing possibilities in reducing security and privacy issues.

This work may be extended for recovery of destructed files if it is further needed. Moreover this system may be adapted for dealing with Big Data analysis with slight modifications.

## REFERENCES

[1] Xuefeng Liu, Yuqing Zhang, "Mona: Secure Multi-Owner Data Sharing for Dynamic Groups in the Cloud", IEEE Transactions on Parallel and Distributed Systems, vol. 24, No. 6, June 2013.

[2] Shucheng Yu, Cong Wang, Kui Ren, Wenjing Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing", Proc. IEEE INFOCOM, pp. 534-542, 2010.

[3] Dan Bogdanov, "Foundations and properties of Shamir's sharing scheme", Research seminar in Cryptography, University of Tartu, Institute of Computer Science, May 2007.

[4] M. Kallahalla, E.Riedel, R.Swaminathan, Q.Wang and K.Fu, "Plutus: Scalable Secure File sharing on Untrusted Storage", Proc. USENIX Conf. File and Storage Technologies, pp. 29-42, 2003.

[5] V.Goyal. O.Pandey, A.Sahai, and B.Waters, "Attribute – Based Encryption for Fine-Grained Access Control of Encrypted Data", Proc. ACM Conf. Computer and Communication Security (CCS), pp. 89-98, 2006.

[6] D.Boneh, X.Boyen, and H.Shacham, "Short Group Signature", Proc. Int'l Cryptography Conf. Advances in Cryptography (CRYPTO), pp. 41-55, 2004.

[7] A. Shamir, "How to Share secret", Comm. ACM, vol. 22, No. 11, pp. 612-613, 1979.

[8] Kyle Chard, Kris Bubendorfer, Simon Caton, Omer F. Rana, "Social Cloud Computing: A Vision for Socially Motivated Resource Sharing", IEEE Transactions on services computing, vol. 5, no. 4, October, 2012.

[9] Lingfang Zeng, Shibin Chen, Qingsong Wei, and Dan FengWuhan**,** "SeDas: A Self-Destructing Data System Based on Active Storage Framework", IEEE Transactions on Magnetics, vol. 49, no. 6, June, 2013.

[10] Jason croft, Robert signorile, "A self destructing file distribution system with feedback for peer to peer networks", Proc. ACM Conf. Computer and Communication Security (CCS), pp. 97-106, 2011.

[11] R. Geambasu, T. Kohno, A. Levy, and H. M. Levy, "Vanish: Increasing data privacy with self-destructing data," in *Proc. USENIX Security Symp.*, Montreal, Canada, Aug. 2009, pp. 299–315.

[12] Roxana Geambasu, Tadayoshi Kohno, Arvind Krishnamurthy, Amit Levy, Henry Levy, "New Directions for Self-Destructing Data Systems" University of Washington Paul GardnerVuze, Inc.Vinnie Moscaritolo PGP Corporation.

[13] R.Perlman, "The Ephemerizer: making data disappear", journal of information system security, 2005.

[14] Prashant Pilla, "Enhancing Data Security by Making Data Disappear in a P2P Systems", Computer Science Department journal, November 2012, Oklahoma State University, Stillwater.

## BIOGRAPHIES

Mr.Ranjith.K received the B.E-Computer Science and Engineering Degree from Sri Jayaram Engineering College affiliated to Anna University in 2011 and currently pursuing his M.Tech-Information Technology Degree in V.S.B Engineering College, affiliated to Anna University. His area of interest includes Cloud Computing, Distributed Systems and Computer Networks.

Mr.P.G.Kathiravan received M.Tech (Information Technology) from K.S.Rangasamy College of Technology (Autonomous) in 2012, Anna University, Chennai and B.Tech (Information Technology) from Kavery Engineering College in 2010, Anna University, Chennai. He worked as Lecturer in the Department of Computer Science and Engineering & Information Technology, Government College of Technology, Coimbatore in the year (2012-13).He is currently working as Assistant Professor in the Department of Information Technology in V.S.B Engineering College, Karur .He published and presented various International & National papers and attended various International & National Workshops. His area of interest includes Computer Networks, Cloud Computing, and Grid Computing.