Privacy Preserving by Self-Destructing Data Using Time Factor in Cloud

Prabhavathi, Geetha. N, Jyothi. T. V
M.E/M.Tech, Department of CNE,
Visvesvaraya Technological University, Belgaum.

*Abstract:***With rapid emergence of cloud computing and internet, services provided by cloud is playing major role. Data stored in cloud may contain some personal and confidential information, which can be misused by miscreants. These data are cached and archives by service providers often without user's control. Privacy is the main issue in cloud, where data stored on cloud may sometimes be leaked due to miscreants or by negligence of service providers. Self-destructing aims at providing privacy to user data. Data becomes self-destructed or unreadable after user specified time. Decryption keys associated with data is also destructed after user specified time.**

Keywords: Self-destructing data, data privacy, active storage, cloud computing.

I.INTRODUCTION

Cloud computing is that the promising utility computing, where applications and services area unit getting into the Web is referred to as "cloud". The cloud users store their resources into the cloud servers and obtain the number of your time they use the services. Several firms like Amazon, Google, SUN, and IBM have empowered in cloud computing and offers cloud based solutions.

The data storage in Cloud and in P2P networks is different from that of data stored on personal machines. Data stored in Cloud and in P2P networks is distributed over many servers and could be compromised at any time if it is not properly secured. Trusting the Cloud and P2P systems for securing confidential data is risky. Simple way or solution is to encrypt the data and store it in a database to avoid archiving and caching. But, even after encryption the data can be decrypted by the cloud service provider, because the service provider has access to all the data, and even to the keys and also data may have been cached. A simple encryption is not enough to keep the data secure.

Data stored in cloud may contain account numbers, passwords and some confidential data. With popularity of cloud computing and internet, people have started relying more on services provided by both. People are requested to provide private and personal information to cloud through internet. Moreover, people hope that security policy will be taken care by cloud service providers and protects their data from leakage.

When people upload their data to cloud, data will be copied, archived and cached for better network performance. This is done without any intimation to user and so user will not have control over it, which may lead to their privacy leakage. Sometimes privacy can be leaked via negligence by service providers and hacker's intrusion.

Study of vanish, provides idea for sharing and protecting privacy. In this system, the secret key is divided into equal parts and is stored in p2p network with distributed hash tables (DHTs).According to p2p characteristics, after every 8 hours nodes get refreshed by DHTs, which means that key stored in the nodes will get refreshed. With Shamir Secret sharing algorithm, if one doesn't get enough parts of the key, he will not be able to decrypt the data, encrypted with this key, which means that key is destroyed completely.

Some attacks to characteristics of p2p are challenges of vanish, uncontrolled survivability of keys and vanish system is prone to Sybil attacks. Considering these disadvantages of vanish, a new system is designed, a self-destructing system, which is based on active storage framework.

II.OBJECTIVES

This system defines two modules, a self-destruct method object, which is associated with each key part and survival time of each part. This system meets the requirement of self-destructing data with controlled survivable time of key. Objectives are summarised as follows:

1) Shamir's algorithm is used as core algorithm to implement client's distribution keys, where this algorithm divides the keys into equal parts and stores in various storage nodes

2) Using active storage framework concept, object based storage interface is used to store and manage equally divide keys.

3) This work supports security erasing files and random encryption keys stored in a hard disk drive (HDD) or solid state drive (SSD), respectively.

IJERT

III.RELATED WORK

A. Existing System

Vanish, a system that leverages the services provided by decentralised global scale infrastructures, particularly Distributed Hash tables(DHTs).In the vanish system, whenever user uploads file, a random encryption key is generated which is used for encrypting the uploaded users data. The local copy of this key is then destroyed and sprinkled over many p2p nodes with distributed hash tables (DHTs).After a particular time, nodes will refresh, where in keys associated with each node is now completely destroyed. Shamir secret key generation is used for splitting of keys. When one cannot get enough parts of the key, data cannot be decrypted at all. It means data remains safe from the attackers.

Uncontrolled on survivability of key is the major drawback of this system, as refreshment of nodes is fixed for 8 hours. Some special attacks to characteristics of p2p nodes like Sybil attack is also another challenge to this system.

FADE, which is File Assured Deletion, is another approach where it focuses on protecting deleted data with policy based file assured deletion. It is built using standard cryptographic techniques, such that it encrypts the outsourced files to provide integrity and privacy, most importantly it assuredly deletes files making it unrecoverable to anyone.

B.Proposed System

Self-Destructing data system mainly aims at providing privacy to users data by self-destructing data after user specified time.This system overcomes the drawback of vanish,where it was not possible to control the survivable time of the key.Here,the time is under the control of user and the data and its associated decryption keys are completely destroyed after a specified time.without having the decryption keys,data can never be decrypted back,hence data remains safe from attacks.

This system makes use of shamir secret key sharing algorithm,where encrypted key is divided into equal parts and stored in storage nodes.

SHAMIR SECRET SHARING ALGORITHM:

This algorithm is created by Adi shamir.This is a form of secret sharing,where a secret which may be a key is divided into equal parts,giving each node or participant its own unique part,where some part or whole is needed to reconstruct the secret or the key.

*Definition:*This algorithm divides secret S into n pieces of data,say D1.....Dn in such a way that:

1) Knowledge of any k or more D_i pieces makes S easily computable.

2) Knowledge of any $k-1$ or fewer D_i pieces leaves S completely undetermined.

This scheme is called (k, n) threshold scheme. If $k = n$ then all participants are required to reconstruct the secret.

IV .SYSTEM ARCHITECTURE

The fig below shows the system architecture of proposed system:

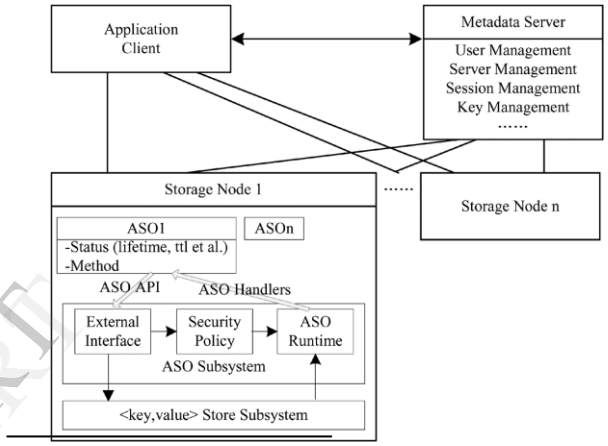

Fig 1.Self-Destructive System Architecture

A. Self-Destructive Architecture

Fig 1 shows the self-destructive architecture, which mainly has three parts associated with it:

1) Metadata Server: Metadata contains data about the data. Here it contains user management, which has the information related to user like user name, address, age etc. It has session management, which keeps track of logins and logouts. Server management is related to data stored in the server and associated keys. Key management keeps track of keys used for encryption and decryption.

2) Client: Client is the user who uses the storage services

3) Storage node: Each storage node consists of two core subsystems i.e., key value store subsystem and an active storage object runtime subsystem. The key value store subsystem is based on object storage component and is used for managing the objects which are stored in storage node like lookup object, read/write object and so on. Here the Object ID itself is used as a key and the associated data along with attribute of the node are stored as values. The

IJERT

active storage object runtime subsystem which is based on the active storage agent module is used to process active storage request from users and manage method objects and policy objects.

B. Active Storage Object (ASO)

An ASO is derived from user object which has time-to-live (ttl) property. This ttl value is used to trigger the self-destruction operation. The time-to-live value for user object will be infinite so that user object will not be deleted until user deletes it manually. The time-to-live value associated with active storage object is limited, so an object will be deleted when the value of the associated policy object is true

C. Self-Destruct Method Object

This method object is a service method. It needs three arguments*: lun* argument which specifies the devices; the *pid* argument specifying the partition and the *obj_id* argument which specifies the object to be destructed.

D. Data Process

In order to use the Self-destructive system, user's applications should implement logic of data process and act as a client node. There are two different logics: uploading and downloading.

1) Uploading Process: Fig 2 shows the file uploading process. When a user uploads a file to a storage system and stores his key in this self-destructive system, he should specify the file, the key and *ttl* as arguments for the uploading procedure. We assume data and key has been read from the file. The algorithm used for ENCRYPT procedure is a common random key generation algorithm or user-defined encrypt algorithm. When the data is uploaded with key and ttl value to the storage server, key shares are generated using Shamir secret sharing algorithm and will be used to create active storage object (ASO) in storage node in the Self-destructive system.

2) Downloading file process: User must have relevant permissions to download the data stored in data storage system. The data has to be decrypted before use. If the ttl value associated with the data has expired, then user reads the data in an encrypted form as the decryption keys are destroyed after time expiration.

When the ttl value is expired, the data which is cached and stored in all the locations gets destroyed or becomes unreadable. One cannot recover the original data due to non availability of decryption key.

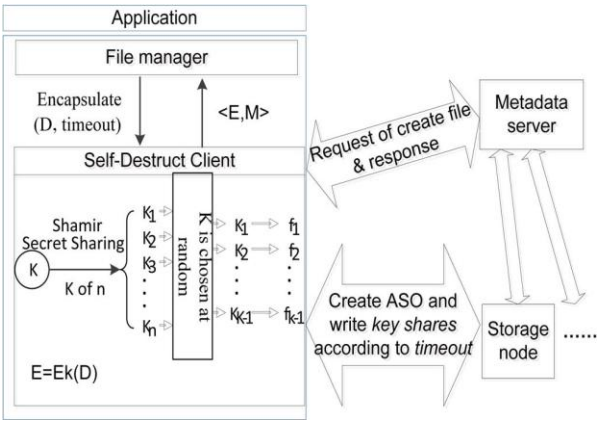

Fig 2: File Uploading Process

E. Data Security Erasing in disk

The sensitive data must be securely deleted in order to reduce the negative impact of OSD performance due to deleting operation. There is no greater proportion of required secure deletion of all the files, so if these parts of the file update operation changes, then the OSD performance will be impacted greatly.

Our implementation method is as follows:

i)In order to store sensitive files, the system pre-specifies a directory in a special area.

ii) To monitor the file allocation table and acquire and maintain a list of all sensitive documents, and the logical block address.

iii) In Logical block address list of sensitive documents appear to increase or decrease, the update is sent to the OSD.

iv) OSDs internal synchronization maintains the list of logical block address, and data in the list updates.

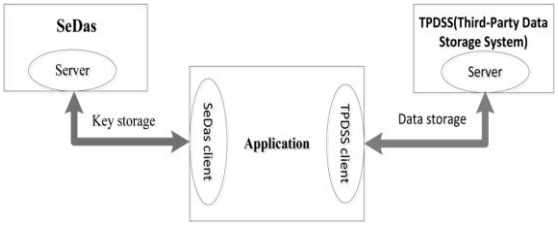

Fig 3.Structure of user application program realizing storage process

V. CONCLUSION

Data privacy in the Cloud environment has become increasingly important. Self-Destructive system is a new approach for protecting data privacy from attackers who will retroactively obtain, through legal or other means, a user's stored data and private decryption keys. This system causes any sensitive information like account numbers, passwords or any secret information to self-destruct after a user-specified time, where there is no any explicit action from user. After specified time the keys associated with data gets destructed, so data will always remain safe from the attackers.

IJERT

REFERENCES

[1] 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.
[2] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[3] S. Wolchok, O. S. Hofmann, N. Heninger, E. W. Felten, J. A. Halderman, C. J. Rossbach, B. Waters, and E. Witchel, "Defeating vanish with low-cost sybil attacks against large DHEs," in *Proc. Network and Distributed System Security Symp.*, 2010.
[4] L. Zeng, Z. Shi, S. Xu, and D. Feng, "Safevanish: An improved data self-destruction for protecting data privacy," in *Proc. Second Int. Conf. Cloud Computing Technology and Science (CloudCom)*, Indianapolis, IN, USA, Dec. 2010, pp. 521–528.
[5] L. Qin and D. Feng, "Active storage framework for object-based storage device," in *Proc. IEEE 20th Int. Conf. Advanced Information Networking and Applications (AINA)*, 2006.
[6] Y. Zhang and D. Feng, "An active storage system for high performance computing," in *Proc. 22nd Int. Conf. Advanced Information Networking and Applications (AINA)*, 2008, pp. 644–651.
[7] T. M. John, A. T. Ramani, and J. A. Chandy, "Active storage using object-based devices," in *Proc. IEEE Int. Conf. Cluster Computing*, 2008, pp. 472–478.
[8] A. Devulapalli, I. T. Murugandi, D. Xu, and P. Wyckoff, 2009, Design of an intelligent object-based storage device [Online]. Available: http://www.osc.edu/research/network_file/projects/object/papers/istor-tr.pdf
[9] S. W. Son, S. Lang, P. Carns, R. Ross, R. Thakur, B. Ozisikyilmaz, W.-K. Liao, and A. Choudhary, "Enabling active storage on parallel I/O software stacks," in *Proc. IEEE 26th Symp. Mass Storage Systems and Technologies (MSST)*, 2010.
[10] Y. Xie, K.-K. Muniswamy-Reddy, D. Feng, D. D. E. Long, Y. Kang, Z. Niu, and Z. Tan, "Design and evaluation of oasis:An active storage framework based on t10 osd standard," in *Proc. 27th IEEE Symp.Massive Storage Systems and Technologies (MSST)*, 2011.
[11] Y. Tang, P. P. C. Lee, J. C. S. Lui, and R. Perlman, "FADE: Secure overlay cloud storage with file assured deletion," in *Proc. SecureComm*, 2010.
[12] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for storage security in cloud computing," in *Proc. IEEE INFOCOM*, 2010.
[13] R. Perlman, "File system design with assured delete," in *Proc. Third IEEE Int. Security Storage Workshop (SISW)*, 2005.
[14] R. Geambasu, J. Falkner, P. Gardner, T. Kohno, A. Krishnamurthy, and H. M. Levy, Experiences building security applications on DHTs UW-CSE-09-09-01, 2009, Tech. Rep..
[15] Azureus, 2010 [Online]. Available: http://www.vuze.com/
[16] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: A public DHT service and its uses," in *Proc. ACM SIGCOMM*, 2005.
[17] [Online]. Available: http://www.planet lab.org/
[18] J. R. Douceur, "The sybil attack," in *Proc. IPTPS '01: Revised Papers From the First Int. Workshop on Peer-to-Peer Systems*, 2002.
[19] T. Cholez, I. Chrisment, and O. Festor, "Evaluation of sybil attack protection schemes in kad," in *Proc. 3rd Int. Conf. Autonomous Infrastructure, Management and Security*, Berlin, Germany, 2009, pp. 70–82.
[20] B. Poettering, 2006, SSSS: Shamir's Secret Sharing Scheme [Online].Available: http://point-at-infinity.org/ssss/
[21] M. Mesnier, G. Ganger, and E. Riedel, "Object-based storage," *IEEE Commun. Mag.*, vol. 41, no. 8, pp. 84–90, Aug. 2003.
[22] R. Weber, "Information Technology—SCSI object-based storage device commands (OSD)-2," Technical Committee T10, INCITS Std., Rev. 5 Jan. 2009.
[23] Y. Lu,D.Du, andT.Ruwart, "QoS provisioning framework for anOSDbased storage system," in *Proc. 22nd IEEE/13th NASA Goddard Conf. Mass Storage Systems and Technologies (MSST)*, 2005, pp. 28–35.
[24] Z. Niu, K. Zhou, D. Feng, H. Chai, W. Xiao, and C. Li, "Implementing and evaluating security controls for an object-based storage system," in *Proc. 24th IEEE Conf. Mass Storage Systems and Technologies (MSST)*, 2007.
[25] Y. Kang, J. Yang, and E. L. Miller, "Object-based SCM: An efficient interface for storage class memories," in *Proc. 27th IEEE Symp. Massive Storage Systems and Technologies (MSST)*,

IJERT