A Systematic Approach for Secured Cloud Computing Using SEDAS Platform

Anisha Kunjan S
M.Tech Computer Science & Engineering
H.K.B.K College of Engineering
Bangalore, India
anishakunjan@yahoo.com

Neelam Malyadri
Associate Professor, Dept. CSE
H.K.B.K College of Engineering
Bangalore, India
malyadri1978@gmail.com

Abstract*— Cloud computing allows consumers and business to use applications without installation and access their personal files at any computer with internet access. The personal data stored in the cloud may contain passwords, pin numbers and other important information that can be misused by an unlawful person or a court of law. These data are copied, cached and archived by cloud service providers (CSP's), often without users' permission. In this paper, we present a SEDAS (Self-Destructing Data) platform that mainly aims to protect user data concealment. All the data copies become erasure or unreadable and the decryption key also destructed after a user specified time, without any user intervention and File recovery is also possible. SeDas platform meets this challenge through an unconventional integration of cryptographic techniques with Active Storage Framework

Keywords*— Cloud computing, Active Storage Framework, data concealment, self-destructing, cryptographic techniques.

I. INTRODUCTION

With the rapid evolution and popularization of Cloud computing and mobile Internet, the Cloud services have become a part of people's life. Users are often requested to submit their personal information to the Cloud by the Internet. When a user submit his/her personal information they hope that their data is protected from leaking and a sufficient level of security is provided by the Cloud Service Providers, so their privacy is not invaded by others.

As users depend more and on Cloud technologies and Internet regularly, the security of their privacy moves towards more risks. On one hand when user's data are being processed and stored by the current computer or network, the system or network must cache, copy or archive the stored data. But, users are unaware of the copies that are cached by the computer or network, and they cannot control them, so this may leak their privacy. On the other hand, the users' privacy can also be disclosed by the Cloud Service Providers (CSP's) negligence or hackers' intrusion. These problems exhibit difficult challenges to protect users' personal data isolation.

A pioneering study of Vanish [1] gives a new idea for sharing and protection of users' data isolation. In this system, a secret key will be divided and stored in a P2P system with Distributed Hash Tables (DHT's). In the P2P system, after about eight hours the DHT will refresh every node. With Shamir Secret Sharing Algorithm [2], when a person cannot find sufficient parts of the key, he/she will not be able to decrypt the data encrypted with this key, which means the key will be destroyed. There are some special types of attacks to the P2P system are the challenges of Vanish [3], and the key survival time limit is also one of the disadvantages for Vanish system.

A novel approach called SEDAS (Self-Destructing Data) platform, which is based on Active Storage Framework [5]-[10] is presented in this paper for the self destruction of data. It defines three modules, a self-destruct method object that is associated with each secret key part, a survival time parameter for each secret key part and a File recovery object for the secret key. Thus, the data security in the cloud environment is enhanced, the network delay is reduced and File recovery is also achieved.

II. RELATED WORKS

A. *Self-Destruct Data*

Vanish [1] is a system for creating messages that automatically self-destruct after a period of time. It integrates cryptographic techniques with global-scale, P2P, distributed hash tables (DHTs): DHTs discard data older than a certain age. The key is permanently lost, and the encrypted data is permanently unreadable after data expiration. It works by encrypting each message with a random key and storing shares of the key in a large, public DHT. However, Sybil attacks [3] may compromise the system by continuously crawling the DHT and saving each stored value before it ages out.

FADE [11] decouples the management of encrypted data and encryption keys, such that encrypted data remains on third-party (un-trusted) cloud storage providers, while encryption keys are independently maintained by a key manager service, whose trustworthiness can be enforced using a quorum scheme. FADE generalizes time-based file assured deletion (i.e., files are assuredly deleted upon time

expiration) into a more fine-grained approach called policy based file assured deletion.

In [13], the paper describes a system that supports high availability of data, until the data should be expunged, at which time the system makes it impossible to recover the data. This design supports two types of assured delete; where the expiration time is known at file creation, and on-demand delete of individual files.

B. *Active Storage and Object-Based Storage*

In [21], the paper describes an Object-based Storage Device (OSD) which is a computer device, similar to disk storage but working at higher level. Instead of providing a block-oriented interface that reads and writes fixed sized blocks of data, an OSD organizes data into flexible-sized data containers, called *objects*. Each object has both data and metadata.

In [25], the paper is based on Storage Class Memory (SCM) which has become increasingly popular in enterprise systems as well as embedded and mobile systems. However, replacing hard drives with SCMs in current storage systems often forces either major changes in file systems or suboptimal performance, because the current block-based interface does not deliver enough information to the device to allow it to optimize data management for specific device characteristics such as the out-of-place update. To alleviate this problem and fully utilize different characteristics of SCMs, we propose the use of an object-based model that provides the hardware and firmware the ability to optimize performance for the underlying implementation, and allows drop-in replacement for devices based on new types of SCM.

In [28], Traditional client/server file systems (NFS, AFS) have suffered from scalability problems due to their inherent centralization. In order to improve performance, modern file systems have taken more decentralized approaches. These systems replaced dumb disks with intelligent object storage devices (OSDs)--which include CPU, NIC and cache-- and delegated low-level block allocation decisions to these OSDs. In these systems, clients typically interact with a metadata server (MDS) to perform metadata operations (open, rename), while communicating directly with OSDs to perform file I/O (reads and writes).

C. *Encryption Key Erase Bits*

In our proposed system, the erasing of files, which include bits (Shamir Secret Shares [2]) of the Key is not enough to erase a file from their storage media. With flash-based solid state drives (SSD's), the erased file situation is more complex due to SSD's internal architecture [19]. The SSD's work in a very different manner than the platter based HDD's, when it comes to read and write processes on the drive.

The analysis in [30] suggests that verifying analog sanitization in memories is challenging because there are many mechanisms that can imprint remnant data on the devices. Reliably erasing data from storage media (sanitizing the media) is a critical component of secure data management. While sanitizing entire disks and individual files is well-understood for hard drives, flash-based solid state disks have a very different internal architecture, so it is unclear whether hard drive techniques will work for SSDs as well.

In most of the previous work aimed at some special applications, e.g., database, multimedia, etc., there is no general system level self-destructing data in the literature.

The experiments show that the proposed SeDas does not affect the normal use of storage system and can meet the requirements of self-destructing data under a life time by user controllable key.

III. DESIGN AND SYSTEM MODEL

A. *SEDAS Platform Architecture*

Fig.1 shows the architecture of the SeDas Platform. There are three nodes based on the Active Storage Framework. **i)** Metadata server-responsible for user management, server management, session management and file metadata management. **ii)** Application node-it is a client to use the storage device provided by SeDas platform. **iii)** Storage node-each storage node is an OSD. It contains two subsystems :< key, value> store subsystem that is based on object storage component and Active storage Object runtime subsystem that is based on active storage agent module in the Object-based storage system.

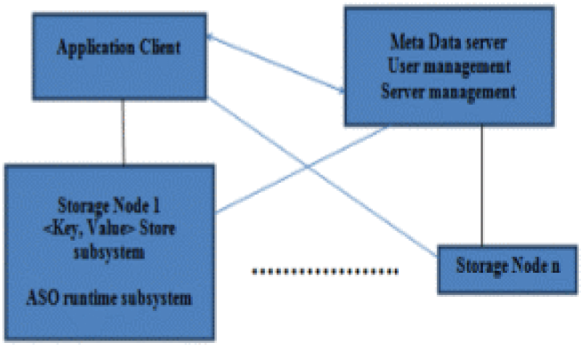


Fig.1 SEDAS Platform Architecture

B. *Active Storage Object*

An active storage object derives from a user object and has a time-to-live (ttl) value property. The value of ttl is used to trigger the self-destruct operation. The ttl value of an active storage object is limited so an active object will be deleted when the value of the associated policy object is true.

C. *Self-Destruct Method Object*

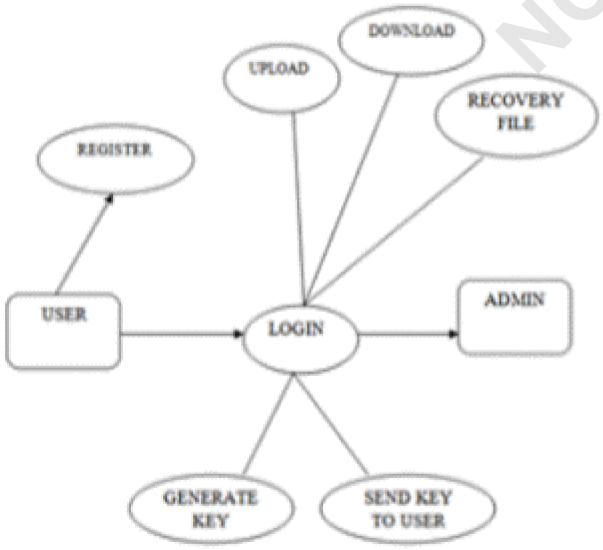
A self-destruct method object is a service method. It needs three arguments. i) *lun* - it specifies the device. ii) *pid* - it specifies the partition. iii) *obj -id-* it specifies the object to be destructed.

D. *Data Process*

In our proposed system, users' applications should implement the logic of data process and should act as a client node. There are three unique logics.

1) *Uploading file process:* When a user uploads a file to a storage system and stores his key in SeDas system, he should specify the file, the key and the ttl as arguments to upload the file.
2) *Downloading file process:* Any user having the relevant access permission can easily download the data stored in the data storage system. The data must be decrypted before use.
3) *Recovery file process:* The data that is destroyed can be recovered by explicitly requesting the Admin to provide the data, which is stored as a backup

The Fig. 2 depicts the three logics as stated earlier.



Data Security Erasing in Disk

We must always secure delete sensitive data and reduce the negative impact of the OSD performance due to the deleting operation. The proportion of required secure deletion of all the files is not so great, so if these parts of the file update operation changes, then the OSD performance will be largely impacted.

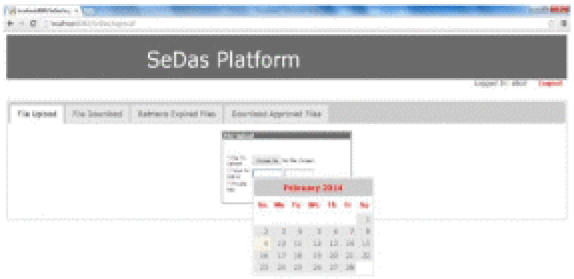
IV. SIMULATION

We have simulated this paper using the JAVA programming language. The simulation consists of 2 roles: System Admin and User. The simulation is conducted in a PC/Laptop with Windows 2007 OS, Editor Software Eclipse, Spring Framework and MySql database.

V. CONCLUSION

Data privacy has become very important in the cloud environment day by day, as more people started relying on the cloud. This paper introduces a new approach for protecting data privacy from attackers who retroactively obtain, through legal or other means, users' stored data and private decryption keys. A novel aspect of our approach is the leveraging of the essential features of Active Storage Framework.

VI. RESULTS



VII. ACKNOWLEDGEMENT

I, ANISHA KUNJAN S, student of M.Tech, Dept., of CSE, H.K.B.K College of Engineering, Bangalore, would like to thank Associate Professor NEELAM MALYADRI for his continuous support and encouragement throughout the completion of this paper. I deeply express my gratitude to all CSE department staff for their valuable suggestion and co-operation.

REFERENCES

[1] 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..

[2] Azureus,2010 [Online].Available:http://www.vuze.com/

[3] 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.

[4] [Online]. Available: http://www.planet-lab.org/

[5] J. R. Douceur, "The sybil attack," in *Proc. IPTPS '01: Revised Papers From the First Int. Workshop on Peer-to-Peer Systems*, 2002.

[6] 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.

[7] B. Poettering, 2006, SSSS: Shamir's Secret Sharing Scheme [Online]. Available: http://point-at-infinity.org/ssss/

[8] M. Mesnier, G. Ganger, and E. Riedel, "Object-based storage," *IEEE Commun. Mag.*, vol. 41, no. 8, pp. 84–90, Aug. 2003.

[9] R. Weber, "Information Technology—SCSI object-based storage commands (OSD)-2," Technical Committee T10, INCITS Std., Rev. 5 Jan. 2009.

[10] Y. Lu,D.Du, andT.Ruwart, "QoS provisioning framework for an OSDbased storage system," in *Proc. 22nd IEEE/13th NASA Goddard Conf. Mass Storage Systems and Technologies (MSST)*, 2005, pp. 28–35.

[11] 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

[12] 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)*,2011.

[13] [Online]. Available: http://www.lustre.org/

[14] B. Welch, M. Unangst, Z. Abbasi, G. Gibson, B. Mueller, J. Small, J. Zelenka, and B. Zhou, "Scalable performance of the panasas parallel file system," in *Proc. 6th USENIX Conf. File and Storage Technologies (FAST)*, 2008.

[15] S. A.Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn, "Ceph: A scalable, high-performance distributed file system," in *Proc. 7th Symp.Operating Systems Design and Implementation (OSDI)*, 2006.

[16] K. Keeton, D. A. Patterson, and J. Hellerstein, "A case for intelligent disks (IDISKs)," *SIGMOD Rec.*, vol. 27, no. 3, Sep. .

[17] M. Wei, L. M. Grupp, F. E. Spada, and S. Swanson, "Reliably erasing data from flash-based solid state drives," in *Proc. 9th USENIX Conf. File and Storage Technologies (FAST)*, San Jose, CA, USA, Feb. 2011