ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

SDD: A Novel Technique for Enhancing Cloud Security with Self Destructing Data

Kishore K, Ramchand V

M.Tech Student, Dept. of CSE, The Oxford College Of Engineering, Bangalore, India

Associate Professor, Dept. of CSE, The Oxford College Of Engineering, Bangalore, India

ABSTRACT: User data on cloud may contain personal information like phone numbers, e-mail id, passwords, ATM pin numbers and etc., which may be misused by miscreants. With growing technology and increasing number of users, the risk to the data also increases. Traditional security techniques include encrypting the user data before uploading which may not work quite efficient for some applications. To overcome this disadvantage, the self destructing data systems came into light. Self destructing system integrates cryptographic techniques to ensure data security. The system will be ensuring two level of security for the user data.

KEYWORDS: Cloud data privacy, Self Destructing data

I. INTRODUCTION

With the growth of the cloud computing, people are turning their existing infrastructure to cloud environment. Otherwise they are adapting to the cloud environment which is a feasible option to reduce the operational cost. As the numbers of users grow on the cloud, the security of the user's data is at more risk. In addition the data uploaded will be cached, archived and stored at many physical locations without user's knowledge. Cloud Service Providers (CSPs) take the responsibility for providing security for these data and their copies, but they may compromise with the security policies over time, which may lead to loss of data privacy. Thus to overcome this situation, it is necessary to have a more secure system to take care of the user's data.

Traditional security systems include encrypting data and uploading the cipher text onto the cloud. This technique was an efficient way of providing security for many decades. The usage of internet grows rapidly and more data is being uploaded to the cloud environment, the security system is at more risk. Introduction of more sophisticated machines to crack a security system makes the existing methods not feasible for many applications. Thus it brings us to search for a new technique to provide security for the user's data.

Self Destructing data refers to the deletion of data after a particular period of time. Self-destructing data mainly aims at protecting the user's data privacy. The entire user's data and their copies become destructed after a specified time, without any user communications. In addition, the decryption key is also destructed after the user-specified time. Self destructing the data in a local system is easy. Destruction includes deletion of a file from a directory. In a cloud environment, one has to delete all the copies of data and it is difficult to know where these copies are stored, as cloud environments are managed and controlled by external entities.

The rest of this paper is organized as follows. We review the related work in Section II, Section III will describe the Design and Implementation of the proposed system. We show some results in Section IV, and conclude the paper in Section V.

II. RELATED WORK

The following section gives the related work so far carried out in the scope of self destructing data.

Vanish [2], is the system that provides the basic idea of self destructing data. The system developed is a prototype which is implemented using two main Distributed Hash Table (DHT). It used bittorrents Vuze DHT that can support eight hours timeout or PlanetLab hosted OpenDHT that can support one week timeout. The system provides a plug-in for Firefox browser that creates a message which automatically disappears after a specified period of time. Here the expiry time for the data is controlled by the DHT and not by the user. Later many extensions have been implemented on the Vanish system.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

Another system called FADE [4], was proposed by tang et al provides a contribution for the self destructing data by integrating cryptographic techniques. The data will be encrypted before sending it. This system will delete the files and makes them unrecoverable by revoking the file access permissions. Another system called File System Design with assured delete [4] proposes three types of file delete. First is the expiration known at the time of file creation, second is on demand deletion of individual files and third is the usage of custom keys for classes of data.

As given above, many systems have been proposed to implement a self destructing system among which only some provide promising results. The system doesn't have a user controllable data expiration time. They rather have a fixed time for file expiration which is not an efficient approach for the self destructing scenario.

Zeng et al [1] came up with the solution for user controllable expiry time with the integration of cryptography techniques along with active storage techniques based on T10 OSD standards. In this system all the data will be self destructed after a specified time after a without user intervention. This system mainly concentrates on encrypting the data before encrypting and using the Shamir's secret key [3] distribution for the safeguarding the key. The key will be retroactively deleted once the expiry time is reached, that makes the data to unreadable.

III. Design and Implementation

A. *Design*

Figure 1. shows the overall architecture of the system. There are four processing modules accordingly. **i) User Application Node:** Each application node is a client's user interface which deals with registration of new users and validation of existing users based on the data stored on the cloud. **ii) SeDas System:** This component is responsible for the self destruction of data. The input to this module is the user's uploaded file and the expiry time attached with each file. The main goal of this module is to make the file inaccessible for anyone upon the expiry of the TTL. **iii) Metadata Server:** This module is a supporting component for the system that provides user management, file management and others. **iv) Administrator:** For the system to work according to the requirement, a central point of control is required. All user's login permissions and file access permissions will be handled by the administrator. The other main goal of administrator is to recover the expired files which are deleted by any user's mistakes.

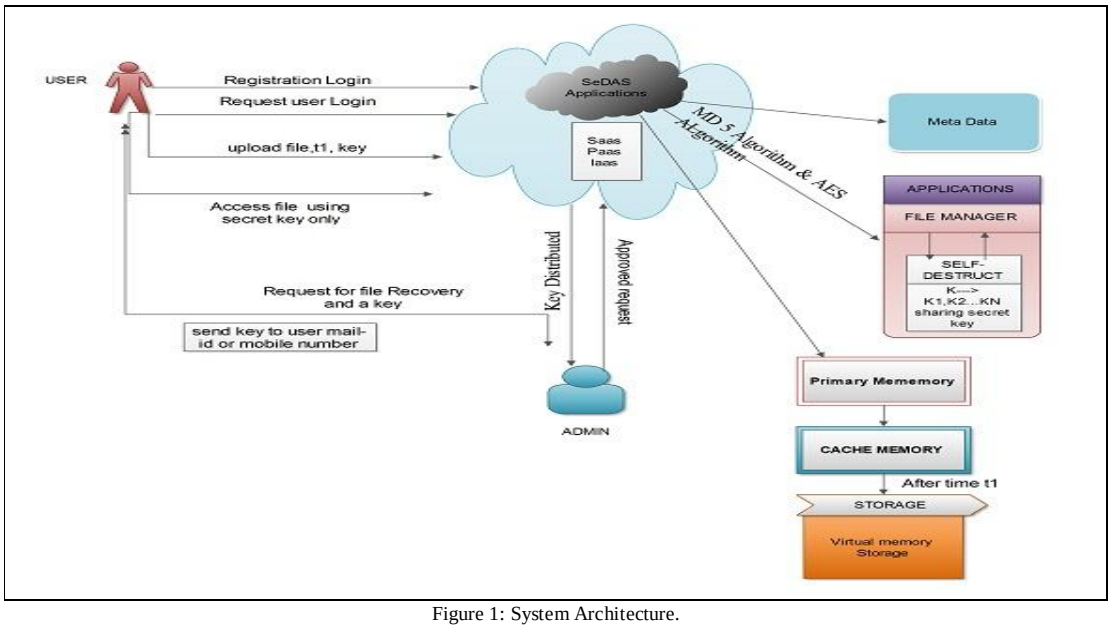

Figure 1: System Architecture.

B. *Implementation*

A prototype for the specified system is developed using the Java technology. The application node includes the options provided to the user for registering to the application. Once the user is registered, the rest of the functionalities like uploading or downloading a file are provided. After the user is authenticated, SeDas system will take the uploaded

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

file by the user along with the key and the TTL parameters as input. The uploaded file will be encrypted using standard cryptographic techniques such as AES algorithm. The key used for encryption will be separated from the source file and will be divided into sub parts using the Shamir's secret key algorithm. These divided parts are then stored under different locations in order to maintain privacy.

The system also appends the TTL parameter to the parts of keys. Once the expiry time is reached the keys will be deleted automatically which makes the data unreadable for anyone. According to the policy of the Shamir's secret key algorithm, if all the parts of keys are not available then it is impossible to decrypt the data and hence the self destruction task is accomplished.

C. Data Processing

The prototype developed is the business logic that works at the client's node. Users have the option of either uploading some data on to the cloud or download the data from cloud. During uploading process, users will attach the TTL to the file. This TTL is different for each file. The file will be encrypted and cipher text will be uploaded to the cloud along with the TTL and the key.

```
PROCEDURE UPDOWN (e-mail, pwd, ph. No)
e-mail: unique e-mail id of the user which works as user id to get registered.
pwd: Secret password gathered from user for login purpose.
ph. No: Valid phone number of the user.

BEGIN
//take the user credentials for registration.
Check the user – id;
if new user then;
        Register the user to the system;
else
        Validate the user with user-id and password;
endif;
// provide the environment for the user to upload and download file.
Check for upload or download option
if upload then
        cipher = ENCRYPT (data, key);
        UPLOAD (cipher, TTL);
else
        if TTL is EXPIRED then
                Send recovery request to administrator and get key;
                DOWNLOAD (cipher);
                System = DECRYPT (cipher, key);
        else
                DOWNLOAD (cipher);
                System = DECRYPT (cipher, key);
        endif;
endif;
END;
```

Figure 2: File Upload and Download (Pseudo code)

The pseudo code for uploading and downloading of files is shown in Figure 2. According to the algorithm specified in our previous paper, if user wants to download the file from cloud, he/she has to check for the expiration of TTL, if the value is expired then the user has to request administrator for the file recovery [11]. The administrator will authenticate the user again and provide the keys for the file to decrypt and use. This feature will help the user to recover his/her files deleted by mistake. If TTL is not yet expired, then the users can download the file along with the key for decryption.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

The job for the administrator is as follows. The administrator on cloud will be constantly monitoring the usage of functionalities provided to the users and the TTL associated with the files uploaded. Once the TTL is expired, the administrator will remove the file from the public location and will make a copy of same in its private location which can be accessed only by him. When a user requests for a file that has been removed from the cloud's public directory, the administrator will in turn authenticate the user and provide the keys and file access permissions to the user with which the user can decrypt the file and use it.

IV. RESULTS

The main goal of the system is to bring the task of self destructing data into reality. The experimental setup includes the usage of the open source cloud service provider such as Jelastic cloud platform. The cloud platform will allow the user to use the basic functionalities and softwares like TOMCAT, JAVA compilers. These services will be provided free for a period of 15 days. This minimum support is enough to deploy our application on cloud. The database provided on the cloud will help to store the files uploaded and the encryption keys used for each file. As the user is not aware of where the data is stored on the cloud, a separate private cloud environment is created and used for the experimental purpose. The following figures shows the results obtained from the observations.

Table 1: Table Showing the Configurations Used for Simulation.

System	Capacity
Processor	CPU: Intel Core 2 Duo, T6670, 2.20GHz
Memory	RAM: 4GB
Internet	Speed: 4Mbps
WEB SERVER	Cloudlets: 4
Platform	Jelastic Cloud Services

.

The above Table 1 shows the basic configurations used at the user system and cloud level for the experimental purpose. Future researchers may use better configurations for a better performance of the system. The algorithm used for the encryption purpose is AES algorithm.

Table 2: Table Showing the Comparison among various encryption algorithms

FACTORS	AES	3DES	DES
Key Length	128,192,256 bits	(k1, k2,k3) 168 bits, (k1, k2 is same) 112 bits	56 bits
Block Size	128,192, 256 bits	64 bits	64 bits
Security	Considered Secure	Stronger that DES, but proven weak.	Proven Inadequate
Possible Keys	2^{128}, 2^{192}, 2^{256}	2^{112}, 2^{168}	2^{56}
Possible ASCII Printable character keys	95^{16}, 95^{24}, 95^{32}	95^{14}, 95^{21}	95^{7}
Time required to check all possible keys	For a 128 bit key: $5*10^{21}$ years.	For a 112 bit key: 800 days.	For a 56 bit key: 400 days.

Table 2 demonstrates the comparison among the available private key cryptographic algorithms. The comparison shows that the AES algorithm is the best available algorithm for this scope of work and hence the algorithm is used accordingly [10].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

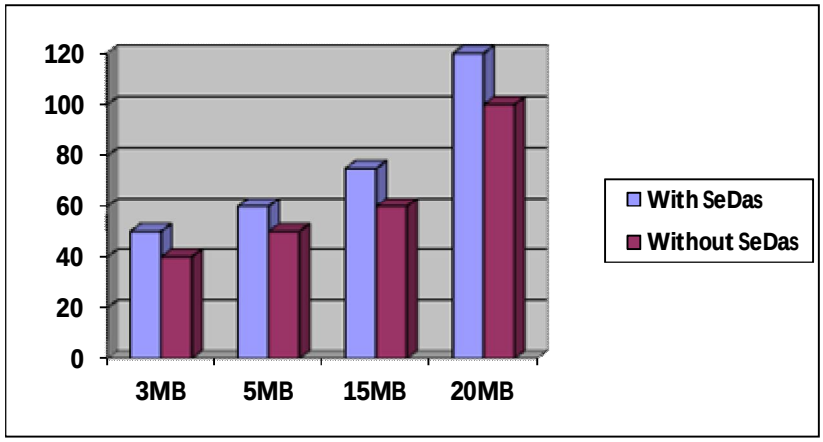

Figure 4: Comparison of Time Taken to Upload a File with different sizes.

The above Figure 4 shows the time taken for the system to upload the file with different sizes. The result shows that there is a small delay taken in SeDas system. This delay is due to the encryption of the file before uploading the file. This delay is negotiable because with increasing internet speed this delay will be lowered with time.

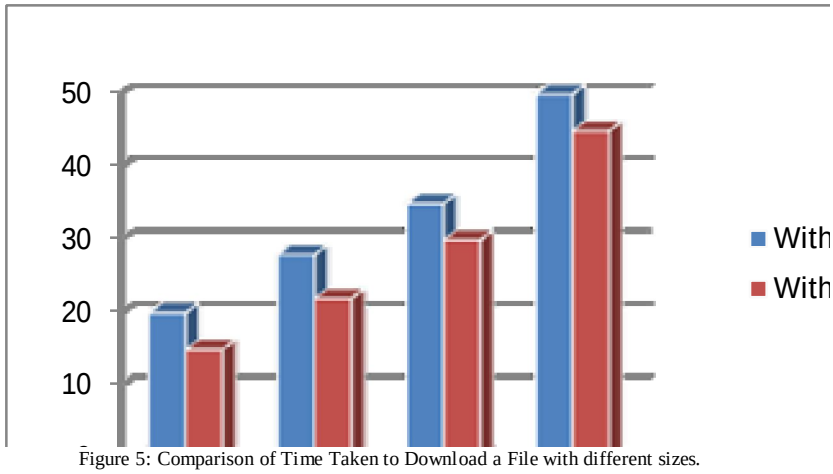

Figure 5: Comparison of Time Taken to Download a File with different sizes.

The above Figure 5 shows the comparison between file different systems for the time taken to download a file. As shown clearly that the difference between the two systems is negligible, the system works faster in the downloading process. And hence this time factor does not make much impact on the acceptability of the system.

V. CONCLUSION AND FUTURE WORK

With growing technology and users, the privacy to the data is at heavy risk which brought the researchers to find a new technique for providing security to the data. In this paper a new technique is introduced which integrates self destructing data concepts and standard cryptographic techniques. This approach provides the latest functionalities in its scope. The system is also feasible to use in a standard cloud environment. The experiment will serve as a base for the future researchers in the scope of self destructing data. Future work will include the integration of additional algorithms for encrypting video files and other formats. This will increase the scope of the application.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 4, April 2014

REFERENCES

1. L. Zeng , S. Chen , Q. Wei , and D. Feng, 'SeDas: A Self-Destructing Data System Based on Active Storage Framework', in Proc. IEEE TRANSACTIONS ON MAGNETICS, VOL. 49, NO. 6, JUNE, 2013.
2. 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.
3. A. Shamir, 'How to share a secret,' Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979.
4. R. Perlman, 'File system design with assured delete,' in Proc. Third IEEE Int. Security Storage Workshop (SISW), 2005.
5. 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.
6. C. Wang, Q. Wang, K. Ren, and W. Lou, 'Privacy-preserving public auditing for storage security in cloud computing,' in Proc. IEEE INFOCOM, 2010.
7. 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.
8. B. Poettering, 2006, SSSS: Shamir's Secret Sharing Scheme [Online]. Available: http://point-at-infinity.org/ssss/
9. 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.
10. Hamdan.O.Alanazi, B.B.Zaidan, A.A.Zaidan, Hamid A.Jalab, M.Shabbir and Y. Al-Nabhani, 'New Comparative Study Between DES, 3DES and AES within Nine Factors', in Proc. Journal Of Computing, Volume 2, Issue 3, March 2010, Issn 2151-9617.
11. Kishore K, Ramchand V, 'A Novel Technique For Enhancing Cloud Security using SeDas Platform', in Proc. International Conference on Convergent Innovative Technologies -2014