International Journals of Advanced Research in Computer Science and Software Engineering
ISSN: 2277-128X (Volume-9, Issue-4)

Research Article | **April 2019**

Self Destructive Data Security System: Towards an Improved Platform for Data Protection

Akpan, Abasiama G
Department of Computer Science, Evangel University, Akaeze – Nigeria
gakpan76@gmail.com

Umeakuka, Chinelo Victoria
Department of Computer Science, Ebonyi State, Abakaliki – Nigeria
chinel2006@gmail.com

Abstract: ***Today's technical and legal landscape presents formidable challenges to personal data privacy. Data protection attempts to ensure the security of computer-processed data from unauthorized access, from destructive user actions and from computer failure. This paper validates a self destructive data security system using files of different formats and a secure server to make an improvement on the existing system. All the data and their copies become destroyed or unreadable when an attempt to copy the file from the secure server is made. This is implemented using php programming language and a prototype methodology. The performance evaluation of the system shows a faster way to upload and download data to be stored and encrypted on the server.***

Keywords: Cryptographic technique, Privacy, Data Destruction, Vanish, Drop box

I. INTRODUCTION

Today's computing environment brings about challenges to personal data and privacy. First, the increasing dependence on web services allows for personal data to be copied and archived by other individuals without knowledge or control. Secondly, the way we disclose data has become common as a result of carelessness, theft, or legal actions. This paper presents a system called vanish that meets this challenge through an integration of cryptographic technique with global scale peer to peer and distributed hash tables. The implementation of a proof of concept Vanish prototype to use the million plus vuze bit torrent distributed hash table and the restricted membership open distributed hash tables. The functionality, security and performance properties of vanish are being evaluated experimentally and analytically and shows that it is practical to use and meets the privacy preserving goals.

The aim is to develop an improved self destructive data security system by developing an algorithm for self destroying system, implementing the model using php programming language and to evaluate the performance of the system.

The ability of searching for the best way of selecting proper security for holding data down as remains unresolved due to the ramped used of internet as the major means of research and storage of data from unauthorized user, now we are trying to implement the use of self destructive file that user can only view but cannot transfer the data in that file to solved unauthorized problems.

II. LITERATURE REVIEW

Data Protection

Data protection attempts to ensure the security of computer - processed data from unauthorized access, from destructive user actions, and from computer failure. With increasing use of computer-based information systems, there has been increasing concern for the protection of computer-processed data. Data protection is closely allied with other functional areas. The design of data entry, data display, sequence control, user guidance, and data transmission functions can potentially affect the security of the data being processed. In many applications, however, questions of data protection require explicit consideration in their own right [1]. Data protection must deal with two general problems. First, data must be protected from unauthorized access and tempering. This is the problem of data security. Second, data must be protected from error by authorized system user, in effect to protect user from their own mistakes. This is the problem of error prevention.

Design techniques to achieve data security and to prevent user errors are necessarily different, but for both purposes the designer must resolve a fundamental dilemma. How can the user interface be designed to make correct, legitimate transactions easy to accomplish while making mistaken or unauthorized transactions difficult? In each system application a balance must be struck between these fundamentally conflicting design objectives [2]. Concern for data security will take different forms in different system applications. Individual users may be concerned with personal

privacy, and wish to limit access to private data files. Corporate organizations may seek to protect data related to proprietary interests. Military agencies may be responsible for safeguarding data critical to national security [3]. The mechanisms for achieving security will vary accordingly. Special passwords might be required to access private files. Special log-on procedures might be required to assure positive identification of authorized users, with records kept of file access and data changes. Special confirmation codes might be required to validate critical commands.

Data Sanitization Policy and Guidelines

Different faculties often have a business need to store prohibited or restricted data on their personal computers. Yet when it comes to dispose of these systems, or transfer them to another user, there are few established policies and processes in place to prevent confidential data from being accessible by unauthorized persons. Data sanitization is the process of permanently removing or destroying data stored on a memory devices irreversibly. These devices discussed include memory devices, CD's and DVD's, Palmtops, Pocket PC's and Smart phones. A device that undergoes sanitization has no useable residual data and even advanced tools would not be able to recover erased data. Exceptions tend to be specialized hardware used by large government agencies to recover sanitized data under special extreme circumstance, and these will not be addressed. It is possible to sanitize a single file, a set of files, or an entire disk or device. Sanitization process includes software that completely erases information, a separate hardware device which is connected to the device being sanitized and erases information and or a mechanism that destroys the devices physically so its data can never be recovered [3].

Data Destruction

Data should be appropriately managed across the entire data lifecycle, from capture to destruction. Planning for data destruction is an integral part of a high quality data management program. Data in any of their forms move through stages during their useful life and ultimately are either achieved for later use, or destroyed when their utility has been exhausted. Establishing policies and procedures governing the management and use of data allow an organization to more efficiently and safely protect its data. Data destruction is the process of removing information in a way that renders it unreadable (for paper records) or irretrievable (for digital record) [4].

Because some methods of data destruction are more complicated, time-consuming, or resource intensive than others, it is common to select the method based on the underlying sensitivity of the data being destroyed, or the potential harm they could cause if they are recovered or inadvertently disclosed. For very low risk information, this may mean simply deleting electronic files or using a desk shredder for paper documents. However, these types of destruction methods can be undone, by a determined and motivated individual, making these methods inappropriate for more sensitive data.

Best Practices for Data Destruction

The information guidance should not be considered comprehensive. Many additional technologies and methodologies exist which may or may not apply to your specific needs. While this document provides high level recommendations, the national institute of standards and technology (NIST) provides in-depth guidance and best practices for the implementation of effective methods of data destruction in their guideline for media sanitation. Modern electronic data storage devices are extremely resilient, and data recovery techniques and technology are highly advanced. Data are routinely recovered from media which have been burned, crushed, submerged in water, or impacted from great heights. Therefore, the approach to data destruction in these two scenarios might be different.

Data Destruction Categories

Data should be appropriately managed across the entire data life cycle, from capture to destruction. Planning for data destruction is an integral part of a high data management program. The following are the data destruction categories to be considered.

- ***Clear***

A method of sanitization that applied programmatic, software, software-based techniques to sanitize data in all user-addressable storage locations for protection against simple non-invasive data recover techniques, typically applied through the standard read and write commands to the storage device, such as by rewriting with a new value or using a menu option to reset the device to the factory state.

- ***Purge***

A method of sanitization that applied physical or logical techniques that renders target data recovery infeasible using state of the art laboratory techniques.

- ***Destroy***

A method of sanitization that renders target data recovery infeasible using state of the art laboratory techniques and result in the subsequent inability to use the media for storage of data.

Self-Destructing Data Systems

Control over data lifetime will become increasingly important as more public and private activities are capture in digital form, whether in the cloud or on personal devices. Self-destructing data systems can help users with some preservative control, by ensuring that data becomes permanently unavailable after a time out specified by the user [5].

- ***Threat Model***

Self-destructing data systems seek to prevent retroactive attacks towards sensitive data. In these systems, user point out certain data as important, encapsulate that data in a vanishing data object, and specify a time out for it.

- ***Vanish System***

Vanish encapsulates data with a user specified timeout by:

1. Encrypting the data with a symmetric key that is not shown to the user.
2. Spitting that key into multiple pieces or shares using threshold secret sharing.
3. Scattering key pieces across randomly chosen nodes in a global scale distributed peer to peer system.

Multi-Backend Architecture for Self Destructing Data

Cascade is a framework for composing multiple backend key-storage systems. The system support simple APIs, both for applications and backend systems, and uses a very flexible hierarchical secret shaving scheme that applications could customize to achieve their desired security properties [3].

- ***Design Principles***

Cascade's architecture is guided by three key design principles:

1. **Combine Several Components with different strengths:** In Cascade, on the other hand, we seek to build a system that is as secure as the union of its defenses. Cascade is a unified self destructing data framework for multiple key storage systems or backend.
2. **Apply Defense in Depth and Redundancy:** Relating to the preceding principle, Cascade provides defence in depth under a single model. While the most pronounced value for cascade is obtained when it combines multiple backend that are secure under different adversarial assumptions, it could also combine multiple backend believed to be secure under the same model.
3. **Support future innovation:** Cascade's design must be very extensible so as to allow the inclusion and deployment increment of new key storage backend. Hence, Cascade provides an environment; within which experimental approaches can be deployed while at the same time benefiting from the security offered by other better proven approaches [6].

- ***Cascade Architecture***

At the top of the architecture are self deleting data applications which may include email, messaging, and social networking, file systems, etc. An application interacts with Cascade through the Cascade Application interface, which encapsulates data into a vanishing data object and later encapsulate that data from the vanishing data object. Encapsulation and Decapsulation requests are handled by Cascade's extensible core. The core functions are also on the same principle as the original vanish system, it encapsulates the data, then generates a key, then spilt it into key shares, then scatters those shares and temporarily stores them on random components of a backend storage system, and then deletes all local copies of K.

CASCADE EXTENSIBLE CORE

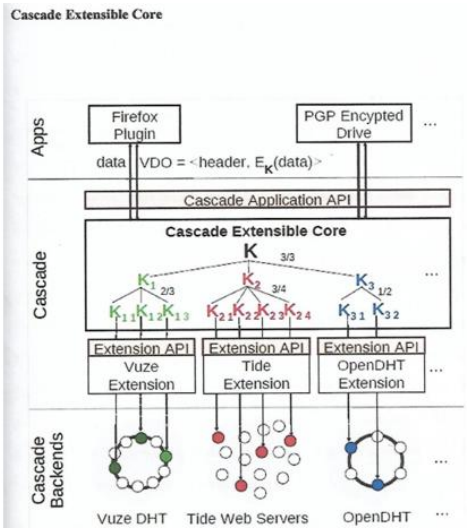

Figure 1.0: Cascade extensible core.

Source: International Jornual on advanced Research in computer science and software Engineering (Saravan et al, 2014).

The Tide Key-Storage System

This section present our second major contribution to self destructing data: Tide, a novel key store for Cascade that combine beneficial properties of both centralized services and decentralized P2P systems: Tide was designed to:

- Be simple, lightweight, and easily deployable,
- Avoid undeleted share residues.
- Avail itself to deployment scenarios that are resilient to malicious infiltration.

The first goal evokes our belief that a small module that is easy to understand, audit, and deploy increases our chances of adoption by many web sites. Tide's ease of development facilitates hosting by end users. Tide web servers to secure their communications.

- **The Vanish Architecture**

A key factor for this work is to leverage existing, de-centralized, large scale distributed Hash Table (DHTs) while a centrally administered system can be compelled to release data by an attacker with legal leverage obtaining subpoenas for multiple nodes.

The Vanish System Architecture

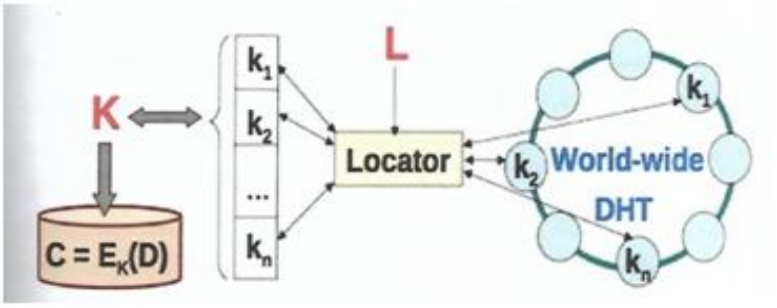

Figure 2.0.: The Vanish System Architecture

Source: A Journal on vanish: Increasing data privacy with self destructing data (Geambasu et al, 2007).

- **Dropbox**

Drop box was developed by Houston and Arash in 2007. It offers cloud storage, file synchronization, personal cloud, and client software. It allows users to create a special folder on their computers, which dropbox then synchronizes so that it appears to be the same folder regardless of which computer is used to view it files placed in this folder are also accessible via the drop box website and mobile apps [7].

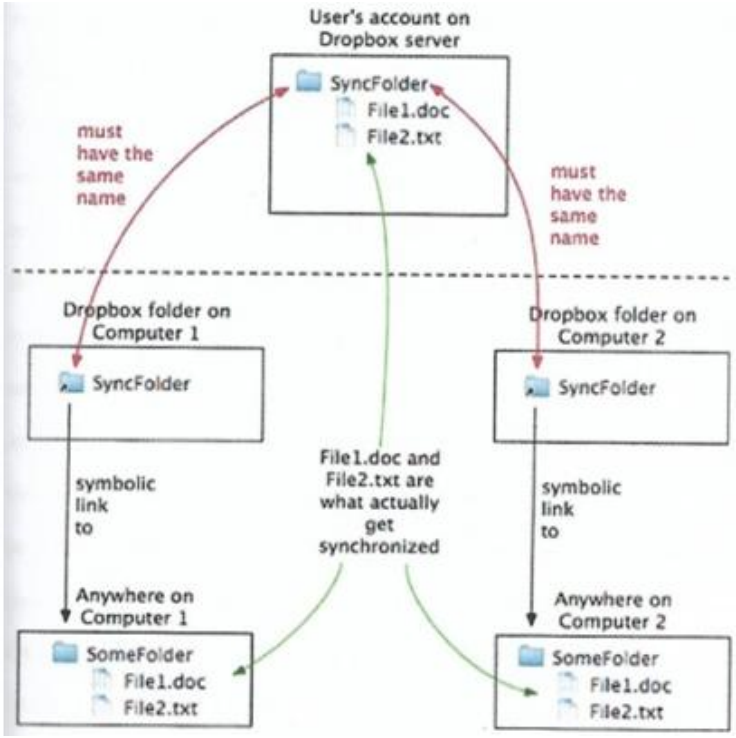

Figure 3:0: Architecture on drop box

Source: A Journal on drop box (Houston et al; 2007)

III. ANALYSIS OF EXISTING SYSTEM

Control over data lifetime will become increasingly important as more public and private activities are captured in digital form, whether in the cloud or on personal devices. Self-destructing data systems can help users preserve some control, by ensuring that data becomes permanently unavailable after a pre-specified time out.

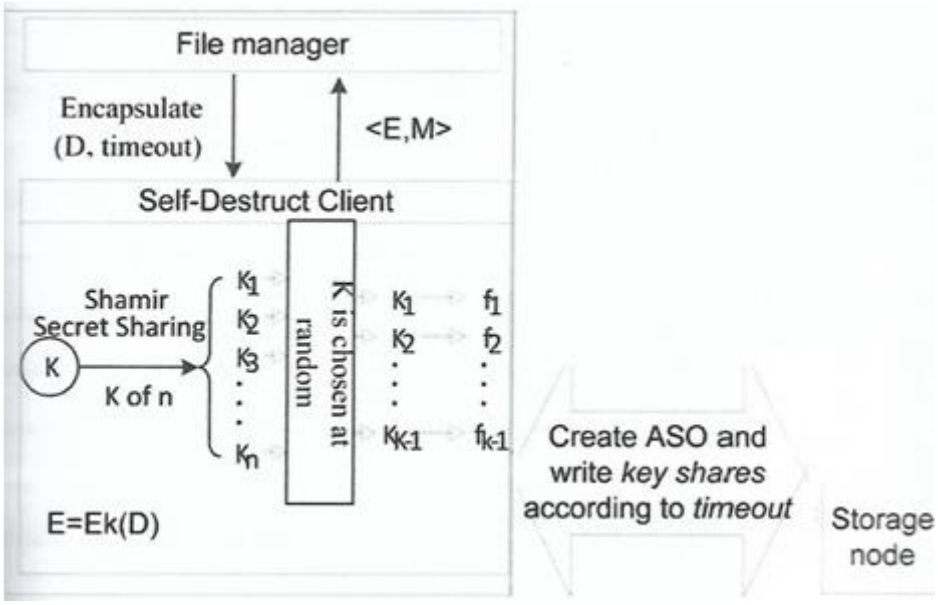

Figure 3.0: Architecture of the existing system

Source: International Journal of software and web sciences (Ramakalpana et. al, 2009).

Analysis of Proposed system

This system tends to create a more secure and rigid way of data security. Unlike the existing system where data is being encapsulated using a pre-specified timeout, this proposed system tends to eliminate the timer application and create a secure server where the data can be accessed through a user specified application. This is a unique way of data security because once the data leaves that server it automatically destroys.

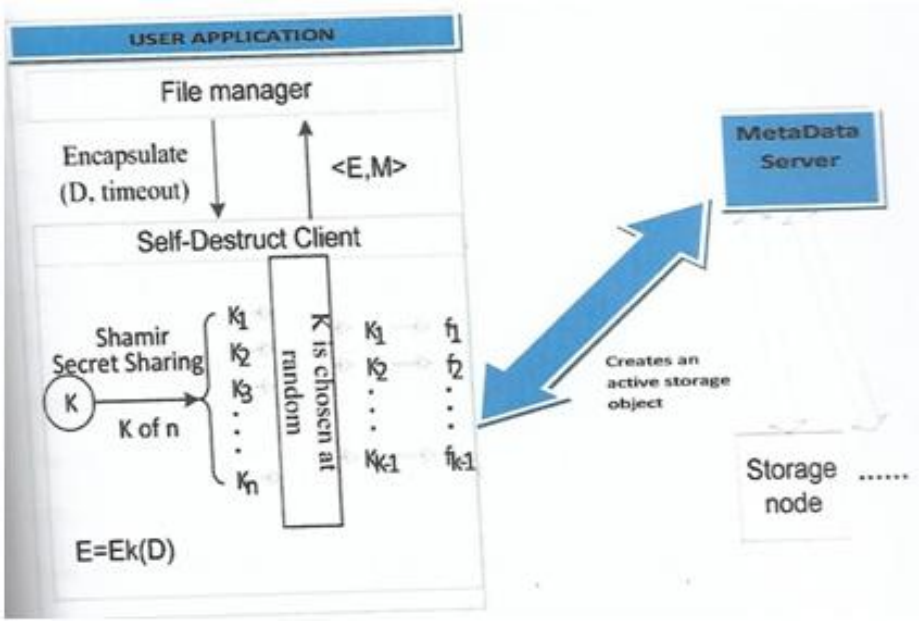

Figure 3.1 (Architecture of the proposed system)

Implementation

We implement the design modeled using the XAMPP technology in the development process. The reason for usage is its simplicity for users. The numbers of systems connected can be many so there is need for extra performance that Apache provides. The technology allows the application to use TCP/IP for internet packet sharing and file/information transfers across the system. The versatility of MySQL and PHP also offers lots of benefits that can help the user in getting information from the internet for proper used since we did host our application by the local host to secure our files.

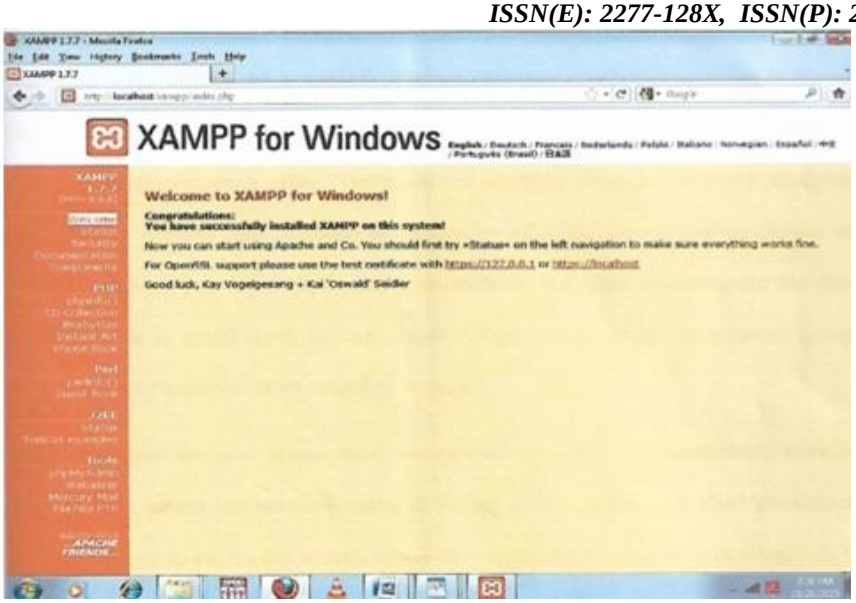

Figure 4.0: Description of a Local host server.

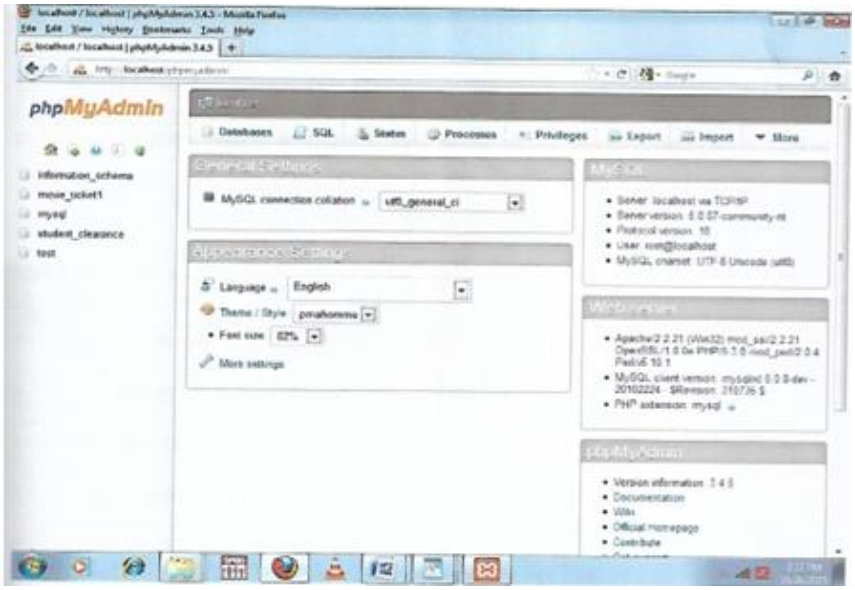

Figure 4.1: MyPhpadmin. Source: www.Xampp.com, 2015

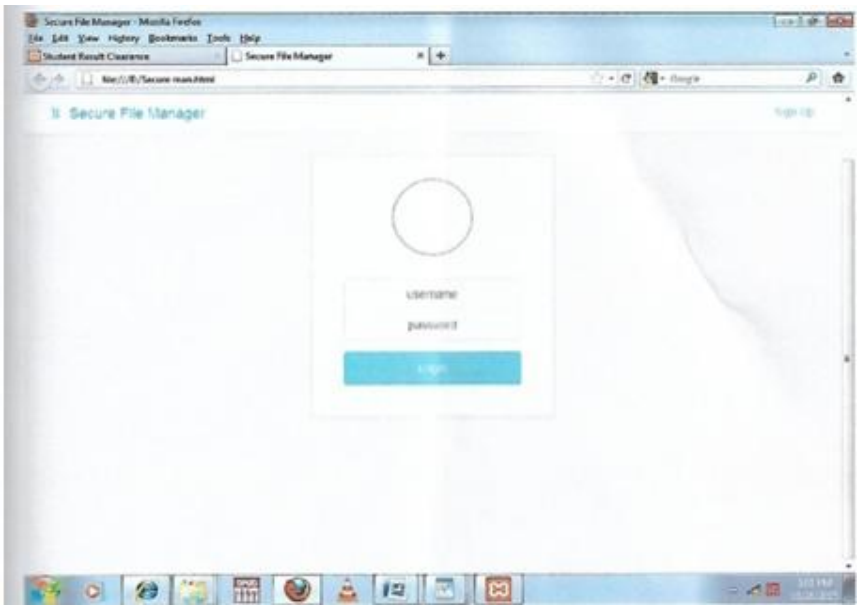

Figure 4.2: The login interface of the new site

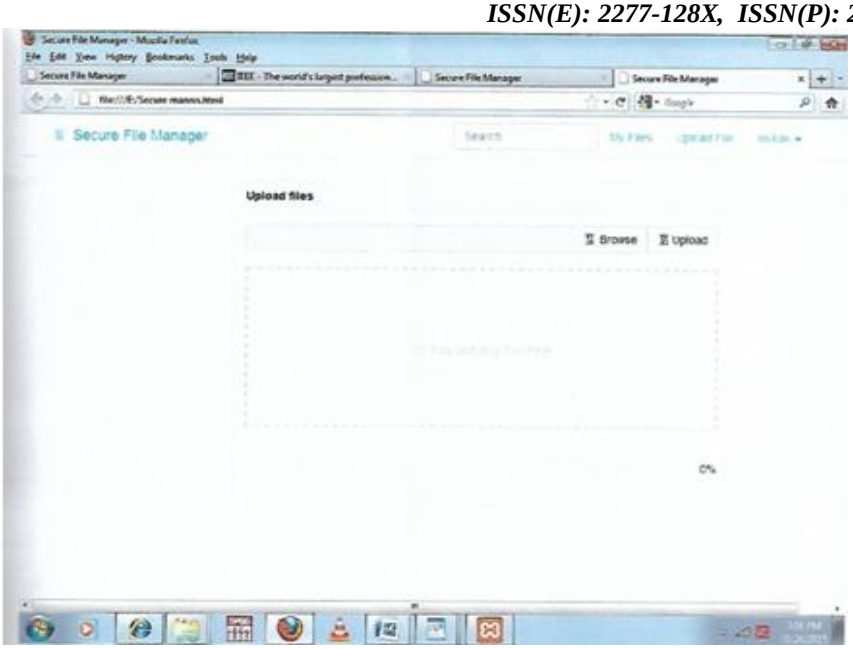

Figure 4.3: Uploading process (using drag and drop process)

Performance Evaluation

The performance evaluation of software is measured with the matrices used. Several matrices can be used in the evaluation of software ranging from speed, time, and efficiency and so on. In self destructive data security system, the matrices used in the evaluation testing are the file size, the time for upload and download, and the throughput.

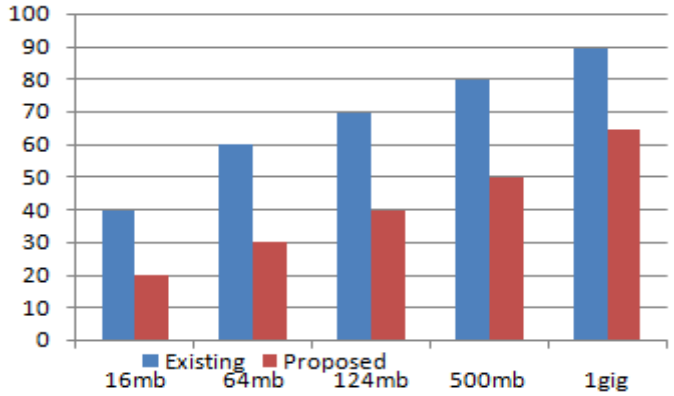

Figure 4.1: Through for upload

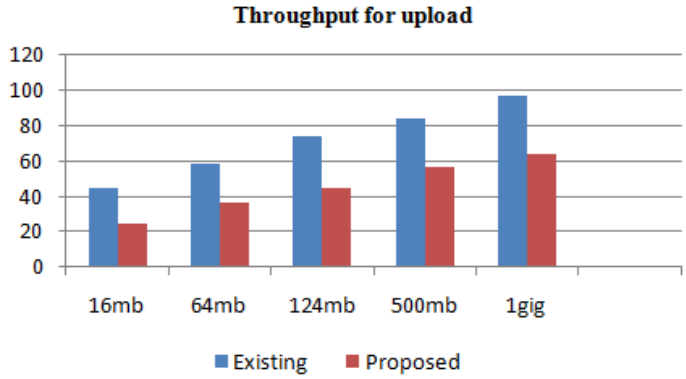

Figure 4.2: Through for Download

Figure 4.1 Shows the through for upload. X*axis* = throughput, Y*axis* = the file size. Throughput is how much megabyte is calculated in one second. In existing system, 16MB was upload in 50s, whereas in proposed system it takes 10s to upload a 16MB file. Also, figure 4.2 shows the throughput for download. X axis = throughput, X axis= the file size. In existing system, 16MB was downloaded in 50s, where as in proposed system it takes 18 sec to download a 16MB.

Result Table

Table 1: performance evaluation uploading

FILE SIZE	EXISTING	PROPOSED
16MB	40s	20s
64MB	60s	30s
124MB	70s	40s
500MB	80s	50s
1GIG	90s	55s

Table 2: performance evaluation for downloading.

FILE SIZE	EXISTING	PROPOSED
16MB	47s	25s
64MB	59s	35s
124MB	73s	45s
500MB	86s	55s
1GIG	96s	65s

IV. CONCLUSION

This paper presented several contributions to the state of self-destructing data systems. We described the Cascade architecture, an extensible framework for integrating heterogeneous key-storage system. We present tide, an Apache-based key-storage system that combines the advantages of DHTs, such as wide-scale distribution, which advantages of centralized systems, such as resistance to crawling attacks. We also presented our extensive experiments with vuze, demonstrating that a security-sensitive design can significantly raise the bar for attackers of DHTs. We believe that this work moves practical self-destructions data systems much closer to reality.

V. RECOMMENDATION

Data privacy is essential in the cloud environment. A new approach is introduced for protecting the data privacy from attackers which may obtain, from legal or other means, a user's stored data and private decryption keys. This system could be recommended for the world as a whole to use to secure data for malicious or unauthorized users.

REFERENCES

[1] Alexander C. and Goldberg I. (2007). An Online Journal on Improved user authentication in off-the- record messaging. In UPES. 5 – 7 accessed 7/5/2015.

[2] Pearlman C. (2005). The Ephemerizer: Making data disappear. Journal of Information System Security, 1(1), 45-78.

[3] Srivatsa M. and Liu C. (2004). An online journal on vulnerabilities and security threats in structured over lay networks: A quantitive analysis. In Proc. Of Annual Computer Security Applications Conference. Vol. 5, 3- 4.

[4] Wolchok M., Hofmann C. Heninger E., Felten H., Halderman A., Rossbach J. Waters M., and Witchel H. (2010). A Journal on Defecting Vanish with low-cost Sybil attacks against large DHEs II in Proc. Network and Distributed System Security Symposium, 2345-340. 30/11/2015

[5] Rabin O. (2005). Provably unbreakable hyper-encryption in the limited access model. In IEEE information Theory workshop on theory and practice in information-Theoretic security. Vol.1, 23-32.

[6] Alexander C. and Goldberg I. (2007). An Online Journal on Improved user authentication in off-the-record messaging. In WPES. 5-7 accessed 7/5/2015.

[7] Sit E. and Morris C. (2002). Security considerations for peer-to-peer distributed hash table. In proceeding of IPTPS. Vol. 7, 11-30.

[8] Drew H. and Arash F. (2007). "An online Journal About Dropbox." 1-4, 16/8/15

