International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

A Survey on Self-Destruction of Data Using Active Storage

Avinash P. Khadse[1], **Prof. Pravin D. Soni**[2]

[1]ME (CSE) Scholar, Department of CSE, P R Patil College of Engg. & Tech., Amravati-444602, India
[2]Assitantant Professor, Department of CSE, P R Patil College of Engg. & Tech., Amravati -444602, India

Abstract: ***With the development of cloud based application, cloud services are becoming more and more important for people's life. People are requested to submit or post some personal private information to the Cloud by the Internet. When people do this, they thought that service provider let secure their private data. By emerging cloud technology maximum applications are using cloud interface, and people usually trust all the cloud services, users generally stores their personal information on cloud. Later that information can be cached, copy, archived by the CSPs' (Cloud Service Provider). Self destructing data is considering as spy proof future of the internet, by using this data is like a time bomb on the network, it means that after certain time interval. Primarily self destruction of data was introduced on DHT (Distributed Hash Table), since low cost Sybil attack on DHT are the problems. Self destructing data is now on active storage framework (SeDas) because it giving higher security and performance. Also uploading and downloading of the file having very poor performance especially with the large files. This paper is going to address these two problems so that privacy can be achieved in public network.***

Keywords: Self Destruction of Data, Cryptography, Active Storage

1. Introduction

Storage is cheap, Data lives forever [2], The Internet never forgets. These truths profoundly affect the interactions between people and modern computing system. Self destruction of data is considered spy proof future of the Internet. With self destructing data privacy can be maintained in the public network. Self Destruction of data goal is to destroy data after certain period of time [1], apart from where the data is stored or archived, in spite of technology that may make such deletion challenging. Consequently, such systems prevent the retrieval of "old" data that is past its useful life[9]. Self destruction is implemented by encrypting data with a key and then retrieving the information needed to reconstruct the decryption key with one or more third parties[12]. Assuming that the key reconstruction information disappears from the retrieval with trust from third parties at the intended time, encrypted data will become permanently unreadable : (1) even if an attacker retrieve a copy of the encrypted data and the user's cryptographic keys and passphrases after the timeout, (2) without the user or user's agent taking any precise action[5] to delete it, (3) with no need to alter any stored or archived copies of that data, and (4) without the user relying on secure hardware.

1. Self-Destructing Data System

Control over data lifetime will become more and more important as more public and private activities are captured in digital form, whether in the cloud or on personal devices[11]. Self-destructing data systems can help users preserve some control, by ensuring that data becomes permanently unavailable after certain period of time.

As people getting more dependent on the Internet and Cloud technology, safety of their privacy takes at very high risks[10]. First when data is being transformed, processed and stored by the current node or network node must cache, copy or archive it. These copies are vital for systems and the network. However, people are unknown about these copies and cannot control them, so these copies may getaway their privacy[12]. And second their privacy also can be leaked via Cloud Service Providers (CSPs') carelessness, hackers' intrusion or some legal actions. These problems present dreadful challenges to protect people's privacy [1].

Limitations of the existing system:

1. SeDas system only deals with text data, no multimedia data is considered.
2. Uploading and downloading of the large files having some performance issue.

2. Literature Survey and Current Scenario

The goal of creating data that self-destructs or vanishes automatically after it is no longer useful. Moreover, it should do so without any explicit action by the users or any party storing or archiving that data, in such a way that all copies of the data vanish simultaneously from all storage sites, online or offline [7]. More generally, self-destructing data is broadly applicable in today's Web centered world, where users' sensitive data can persevere "in the cloud" indefinitely (sometimes even after the user's account termination). With self-destructing data, users can regain control over the lifetimes of their Web objects, such as personal messages on Facebook and documents like Google Docs [3].

Self destructing system can enduringly delete data after certain period of time:

1) Even if an attacker can retroactively get an unspoiled copy of that data and any relevant persistent

cryptographic keys and passphrases from before that timeout, possibly from stored or archived copies.
2) Without the use of any explicit delete action by the user or the parties storing that data.
3) Without needing to modify any of the stored or archived copies of that data.
4) Without the use of secure hardware.
5) Without relying on the introduction of any new external services that would need to be deployed (whether trusted or not).

Self destruction of data was introduced by University of Washington author was Roxana Geambasu et al [2] in Vanish: Increasing Data Privacy using Self Destruction of Data, 2009, in this paper message automatically get "self-destruct" after a period of time and it was done over DHT (Distributed Hash Table). After that in 2009 itself Scott Wolchok et al [1] present two Sybil attacks against the vanish implementation, which supplies its encryption keys in the million-node Vuze DHT. These attacks work by continuously attacking the DHT and saving each stored value before it goes off. They can efficiently recover keys for more than 99% of Vanish messages. SeDas paper shows that the foremost cost of these attacks is network data transfer, not the memory usage as the Roxana expected, and that the total cost is two orders of magnitude less than Roxana estimated[12]. While the consideration is potential defences, Roxana conclude that public DHTs like Vuze [2] probably cannot provide strong security for Vanish. In 2010 SafeVanish was introduced by LingfangZang et al. [4] SafeVanish is based on extending the length range of the key shares and applying the public-key cryptosystem, to multiply the hopping attack cost, including storage requirement and the network bandwidth, and to avoid the sniffing attack[10].

Though Vanish is an interesting approach but it is having important privacy problem. To address the hopping attack problem of Vanish discussed above, LingfangZang et al. [4] proposed a new scheme, called SafeVanish. Hopping attack is one type of the Sybil attacks and it is addressed by extending the length range of the key shares to enlarge the attack cost significantly, and some enhancement on the Shamir Secret Sharing algorithm implemented in the Vanish system[11]. Also, proposed an improved approach not in favour of sniffing attacks by way of using the public key cryptosystem to avoid sniffing operations[8].

Tang et al. [3] proposed FADE which is built upon standard cryptographic techniques and assuredly deletes files to make them unrecoverable to anyone upon revocations of file access policies. Wang et al. [4] utilized the public key based homomorphism authenticator with random mask technique to achieve a privacy-preserving public auditing system for Cloud data storage security and uses the technique of a bilinear aggregate signature to support handling of multiple auditing tasks. Perlman et al. [5] present three types of assured delete: ending time known at file creation, deletion of individual files as require, and formation of custom keys for classes of data.

However, the use of P2P features still is the serious weakness both for Vanish and SafeVanish, because there is a precise attack against P2P methods (e.g., hopping attacks and Sybil attacks)[2].

In addition, for the Vanish system, the endurance time of key realization is determined by DHT system and not convenient for the user. Based on active storage framework, LingfangZang et al. [1] propose a distributed object-based storage system with self-destructing data function. SeDas system combines a positive approach in the object storage techniques and method object, using OSD's data processing capabilities [6] to achieve data self-destruction. User can specify the time to live of the key of distribution and use the settings of stretched interface to export the life cycle of a key, allowing the user to manage the subjective life-cycle of private data.

The whole scenario of the literature survey is given in a table 1 with key points. Key points are describing the resources and features that are hold by respective papers.

Table 1: Literature Survey

Sr. No.	Paper Title	Year of Publish	Key Points
1	SeDas: A Self-Destructing Data System Based on Active Storage Framework.	JUN 2013	Active Storage, Cloud Network, Shamir Key Distribution, TTL (time to live), Metadata Server.
2	Vanish: Increasing Data Privacy with Self-Destructing Data.	AUG 2009	DHT, P2P, Client based appn, VDO (Vanish Data Object), Locator.
3	Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs.	SEP 2009	DHT, P2P, Advance hopping implementation.
4	SafeVanish: An Improved Data Self-Destruction for Protecting Data Privacy.	NOV 2010	DHT, P2P, Hopping attack, expanded range of key shares.

3. Proposed Work

Future versions of Vanish and Vuze could adopt various countermeasures against crawling attacks[7]. While we discuss several strategies for making these attacks more expensive, it seems difficult to raise the cost enough to provide strong resistance without sacrificing other security goals, usability, or reliability.

There are several problems for Vanish security using a smaller-scale DHT, even one with special security features. A DHT with a small user base or a single maintainer is vulnerable to social collusion. In the case of OpenDHT, convincing its single maintainer to add an anti-Vanish feature would compromise the security of Vanish. It is also easier to convince enough participants to subvert the security of a system when the user base is small and drawn from a particular community (e.g., tens of academic users in the case of OpenDHT). Lastly, a privately hosted DHT like OpenDHT would essentially function as a trusted third party, and there are simpler ways to implement Vanish-like behavior in applications where invoking a trusted third party is acceptable.

Adding client puzzles to Vuze Client puzzles have been proposed as a defenses against Sybil attacks[8]. A simple approach would be to require clients to perform an expensive computation tied to the current date and their node ID. For example, if Vuze required a daily computation that took one minute on a small EC2 instance, this would impose a cost of $0.34/year for each Sybil. To obtain 90% VDO recovery, we need 107,000 effective Sybils (for k = 45 and n = 50), so we would need to devote 74 EC2 instances to solving puzzles. This would raise the cost of our attacks by about $37,000 per year. Though this is a significant increase, it only impacts attackers who actually pay for CPU time—an attacker who controlled even a small botnet could easily perform the puzzle computations. In addition, if the puzzles were predictable, then an attacker might use pre-computation to solve several puzzles for a certain time period. While the attacker might not be able to sustain this attack, all VDO created during this time period would be vulnerable.

Detecting attackers another possible defense is to try to detect attackers and selectively block or penalize their interactions with Vuze. One approach would be to monitor peers for deviations from the Vuze protocol that distinguish them from legitimate clients. This is currently easy to do for our ClearView software, which omits certain functionality for ease of implementation, but attackers might try to avoid detection by responding to requests more faithfully. A second approach would be to monitor IP addresses that host an unusual number of Vuze clients. Incrementing the Vuze bootstrap node or scanning the routing tables maintained by peers in the network would detect such IPs.

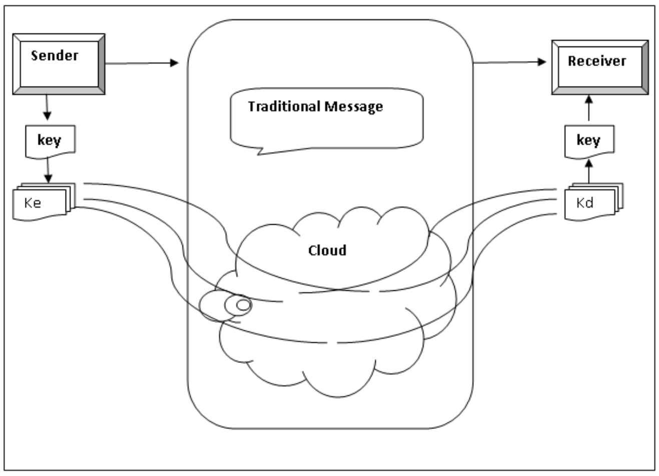

Figure 1: Architecture based on proposed work

The self destructing data system consists following steps:

Sender (Message) -> Message Encryption -> Key slicing-> Key Distribution -> Key Integration / Reconstruction -> Decryption (Message with reconstructed key)

Self destruction is implemented by encrypting data with a key and then escrowing the information needed to reconstruct the decryption key with one or more third parties. Assuming that the key reconstruction information disappears from the escrowing third parties at the intended time, encrypted data will become permanently unreadable.(1) Even if an attacker obtains a copy of the encrypted data and the user's cryptographic keys and passphrases after the timeout, (2) without the user or user's agent taking any explicit action to delete it, (3) without needing to modify any stored or archived copies of that data, and (4) without the user relying on secure hardware.

4. Conclusion and Future Scope

Data privacy has become extremely important in the Cloud environment. In this paper, the methods to solve the issue of self-destruction of data on networks have been summarized. The object interface offers storage that is secure and easy to share across platforms, but also high-performance, thereby eliminating the common trade-off between files and blocks. Furthermore, objects provide the storage device with an awareness of the storage application and enable more intelligence in the device. Although there are many proposed solutions so far, there is no perfect solution yet. Many proposals are still not mature and need further research and experimentation. Besides these proposals, designing novel and suitable network architecture for self-destruction and Active storage and to improve its performance is also a hot research topic. Proposed approach is using active storage framework for maximum advantage and it is expected that after successful implementation, performance should be more than existing systems. With this privacy can be achieved in a public network.

Acknowledgment

We wish to thank the Department of Computer Engineering also we are thankful of Project Lab of P.R. Patil COE for providing all resources and their valuable support.

References

[1] LingfangZeng , Shibin Chen , Qingsong Wei , and Dan Feng, "SeDas: A Self-Destructing Data System Based on Active Storage Framework," Wuhan National Laboratory for Optoelectronics, School of Computers Huazhong University of Science and Technology, 430074 China, 2013.
[2] Roxana GeambasuTadayoshi Kohno Amit A. Levy Henry M. Levy, "Vanish: Increasing Data Privacy with Self-Destructing Data," University of Washington, United States, 2011.
[3] Scott Wolchoky, Owen S. Hofmanny, Nadia Heninger3, Edward W. Felten, J. Alex Halderman, Christopher J. Rossbach, Brent Waters, and Emmett Witchel, "Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs," Department of Electrical and Computer Engineering, Polytechnic Institute of New York University, United States, 2011.
[4] LingfangZeng, Zhan Shi, ShengjieXu, Dan Feng, "SafeVanish: An Improved Data Self-Destruction for Protecting Data Privacy," Wuhan National Laboratory

for Optoelectronics, School of Computers, China, 2010.

[5] Lingjun Qin, Dan Feng, “Active Storage Framework for Object-based Storage Device Key Laboratory of Data Storage Systems,” Ministry of Education of China, E-mail: qinlingjun@yahoo.com.

[6] YulaiXie et al, “Design and Evaluation of Oasis: An Active Storage Framework based on TIO OSD Standard”, School of Computer, Huaz}lOng University of Science and Technology, ylxie@smail.hust.edu.cnthe 1982 conference on Human factors in Computing Systems

[7] KamasaniLokesh , P .Hemanth Kumar, “Novel Framework to Cloud data security” Computer Science Engineering, KMMITS, Tirupathi, Vol 10, July 2014.

[8] A. Shamir, “How to share a secret,” Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979.

[9] Special Publication 800-131A, “Transitions: Recommendations for Transitioning the Use of Cryptographic Algorithms and Key Lengths,” [http://csrc.nist.gov/publications/PubsSPs.html#800-131A]

[10] C. Saravanan , E.Nisha , P. G. Karunya International Journal of Advanced Research in Computer Science and Software Engineering “Cascade Architecture for Self Destructing Data System Using Vanishing Data Object”

[11] R.Rengasamy1, V.Kumaresan2, G.Guru Rani3 “A Self Destructing Data System Based on Active Storage Framework for Protecting Data Privacy from attackers UN agency” R.Rengasawy et al, International Journal of Computer Science and Mobile Computing, Vol.3 Issue.4, April- 2014, pg. 1375-1379 © 2014

[12] Mohan Sadasivam, Rajeeve Dharmaraj “SADS – Self Annihilating Data Storage system in Cloud Storage Service”, International Journal of Information & Computation Technology. ISSN 0974-2239 Volume 4, Number 11 (2014), pp. 1035-1042 © International Research Publications House http://www. irphouse.com

Author Profile

Mr. Avinash P. Khadse received Bachelor degree in Computer Engineering from Nagpur University in 2012 and pursuing master degree in C.S.E from P.R. Patil college of Engg Amravati – 444602.

Prof. Pravin D. Soni is working as Assistant Professor in department of C.S.E. at P.R. Patil College of Engg Amravati -444602.