International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 3 | ISSN : 2456-3307

Security Framework for Cloud Data Sharing

E. Dhivyaprabha[1], R. Madhubala[2], M. Meena Abarna[2]
[1]Assistant professor, Computer Science and Engineering, Sri Krishna College of Technology, Coimbatore, Tamil Nadu, India
[2]Student, Computer Science and Engineering, Sri Krishna College of Technology, Coimbatore, Tamil Nadu, India

ABSTRACT

Cloud computing is a growing technology that offers storage as a service to user where data is maintained, managed, backed up remotely. Security of cloud-based applications and data is one of the key concerns of cloud customers. Some of the proposed cryptographic techniques provide data users with high security in terms of outsource storage; however many of the encryption algorithm are knackered, activating data security to be got through simply by compromising an algorithm. This scheme discovers a combination of encryption algorithms and distribution servers to improve database privacy. The main goal of this work is to develop a secure and efficient auditing scheme with the capabilities such as privacy preservation, confidentiality, and data integrity. So it satisfies all the requirements as well as it reduces cloud server burden.

Keywords: Data security, Auditing, cloud storage, privacy preserving, encryption

I. INTRODUCTION

Cloud computing can be defined as a new flair of computing in which real time scaling and virtualized resources are provided as a services. Providing security for data is a major concern in cloud storage systems even though it comes with attractive benefits [1, 2]. The internal and external threats cause the data in cloud to be deleted or corrupted or tampered [4, 5]. In specific any external adversary tries to alter the content of the stored data and convinces the owner of the data that their data stored in cloud is correct and intact [6]. This is being done for high profit. Hence it becomes essential to verify the correctness and integrity of the outsourced data moved to cloud.

Data privacy, protection of data, data availability, data location, and secure data transmission are some issues that need to be looked up in cloud data security. Threats, data loss, service disruption, outside malicious attacks, and multi tenancy issues are some of the security challenges in the cloud. Data integrity means maintaining the correctness of stored user's data on the cloud. Any unauthorized users should not be able to modify or hack the stored data on the cloud. Cloud computing providers are trusted for maintaining the data integrity and accuracy of the data. Data confidentiality is also an important factor from user's point of view as users tend to store their important and confidential data on the cloud.

Authentication and access control strategies are used to ensure data confidentiality. The data confidentiality issue can be resolved by increasing the cloud reliability and trustworthiness in Cloud computing. Therefore, data security, data integrity, privacy and confidentiality of the stored data on the cloud are the important factors to be considered from user's point of view [2]. To meet this requirement,

new methods or techniques should be developed and implemented.

Data auditing is a new concept introduced in cloud computing to deal with secure data storage. Auditing is a process of verification of user data. It can be carried out either by the user himself (data owner) or by a TPA. It helps to maintain the integrity of data stored on the cloud. Similarly, asymmetric key cryptography involves decryption on the provider's side, allowing providers to infer sensitive information.

The three-network entities viz. the data owner, cloud server and TPA are present in the cloud environment. The data owner is responsible to store data on the storage server provided by the cloud service provider (CSP). TPA keeps a check on the client's data by verifying the integrity of data on demand. It notifies data owner if any variation or fault. Fig.1 shows the cloud data storage architecture.

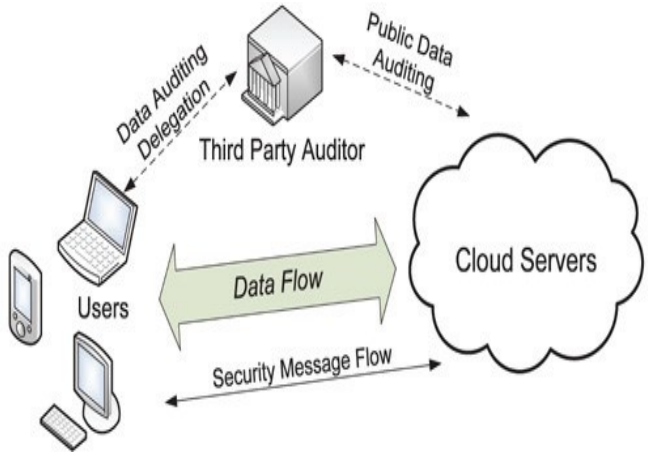

Figure 1. Cloud Data Storage Architecture [8]

II. LITERATURE SURVEY

There are many issues that Cloud computing undergoes on the integrity and privacy of the user's stored data at cloud. It is important from the user's perspective to develop a secure and efficient method to guarantee the integrity and privacy of data stored in the cloud.

(a) Wang et al. [4] has proposed a privacy preserving public auditing protocol. It uses dispassionate TPA for reviewing the data. For this purpose, Homomorphism linear authenticator (HLA) a public key based approach along with random masking techniques is used. Although it is exposed to existential forgeries attacks such as message attack from a malicious cloud server and also an outside attacker. To resolve this issue, Wang et al. [3] Proposed a new improved scheme which tends to be more secure than the previous proposed protocol [4]. It is also a public auditing scheme with TPA. It is developed to perform data auditing on behalf of users. Wang et al. [6] proposed another protocol that supports both public auditing and data dynamics by using the BLS based HLA along with Merkle Hash Tree (MHT). It achieves the integrity of data, but fails to provide confidentiality of the data stored on the cloud. Wang et al. [7] has also proposed a design to detect the modified blocks easily using homomorphism token pre-computation and later erasure coded technique is used to acquire the desired blocks from different servers.

(b)Bouganim and Pucheral [8] have proposed a hardware software solution for the database outsource confidentiality issue, claiming that no software solution is guaranteed, given Internet security variability. Their idea is basically to install a smartcard on the cloud side that will work as a mediator. The data is then encrypted by the smartcard before being inserted into the database, and then decrypted before being sent to the user. This method is premised on an assumption of secure communication between the user and the cloud. The smart card is fully controlled by the client, so the cloud is simply used as encrypted data storage. This approach has limitations in terms of processing power as well as memory capacity and thus does not offer a practical solution for the outsourcing of confidential data due to the weakness of the smartcard and the difficulty of installing the card into the cloud provider's server. The situation

worsens if the database system needs to be distributed.

(c)Popa et al. [4] recently proposed a practical solution to improve the confidentiality level for outsourced data from curious cloud providers. Their scheme consists of a number of components, including encryption algorithms, proxy, and user's application. This concept relies on the fact that no single encryption algorithm can support all types of queries, so the researchers investigated encryption algorithms that allow for queries to be issued to encrypted data. They came up with six encryption algorithms that can be used to support the fundamental query structure. Other studies belonging to this category can be found in [5]–[9]. In the existing systems, the task of computing the proof for integrity check of data is carried out by a cloud server who is also responsible for storing huge amount of user's data. Thus increasing the burden of storage as well as the task of verification proof generating on the cloud server. There is a need to propose a system which does not increase the load on cloud server in the auditing process. All the factors mentioned above are important and need to be achieved for a reliable scheme. Therefore, it is necessary to develop an efficient and secure auditing scheme which can perform public auditing effectively by maintaining both the integrity and confidentiality of stored data.

III. PROPOSED SYSTEM

We propose a combination of encryption algorithms and a distribution technique that together form a novel contribution to this research.

A. Encryption algorithm

The algorithm used in our scheme is Honey Encryption algorithm.

1. Password-Based Encryption

Weak passwords aren't just a problem for hashing; they also impact users ability to encrypt sensitive data using password-based encryption (PBE). PBE carries absolutely the same vulnerability to identifying attacks as hashing PBE basics. A PBE scheme consists of an encryption function enc and a corresponding decryption function dec. A message M is encrypted under a password P as a cipher text C = encP(M). The message can be decrypted as M = decP(C). Given a decryption attempt using an incorrect password P¹P , decP(C)outputs an error message—which we can think of for convenience as a special error symbol ^. A popular standard for PBE is PKCS (Public-Key Cryptography Standard) #5 v2.0.7 (Practitioners often use the key-derivation function PBKDF2 from PKCS #5 v2.0 to derive encryption or decryption keys from passwords. However, advances in cipher and cipher-mode design since the publication of PKCS #5 v2.0 motivate the use of authenticated encryption modes such as EAX over PKCS #5 v2.0 recommendations.)

2. Introducing HE

For some kinds of messages, HE provides security that's beyond the brute-force bound and thus unobtainable with traditional PBE. HE is best-suited for constructions in which encrypted data is derived from passwords. Because each improper guess generates a credible result, it will be hard for the attacker to know when he has guessed correctly. When attackers can try decrypting a cipher text with all possible keys makes attacks unsuccessful by ensuring that all plaintexts generated during a brute-force attack look plausible. This approach was used previously for the special case of uniformly random plaintexts by Kausik and Hoover [6]. Juels and Ristenpart proposed a more general cryptographic primitive that they called honey encryption (HE). A HE scheme is tailored to an estimate of the (possibly non-uniform) distribution of messages for which it will be employed. We refer to this distribution as the target distribution. Decrypting a HE cipher text with an incorrect key yields a decoy (or honey) message that appears, to the attacker, to be a fresh sample from the target distribution. They propose that HE

schemes should achieve security in two distinct settings, what we will call the high-entropy key setting and the low-entropy key setting. The former is the conventional setting in which security rests on the adversary being unable to do work proportional to 2μ. Here they show that DTE-then-Encrypt can use standard mechanisms to provably achieve the conventional goals of [7].

2.1 Distribution-Transforming Encoders

HE models the space of plaintext messages using a distribution-transforming encoder (DTE). Consider pm be a probability distribution over the message space M, meaning that a user selects for encryption with probability pm(M). (We give an example for this intuition later.) A DTE encode encodes M as an ℓ-bit seed S Î {0, 1}ℓ. This encoding needn't be unique: many seeds might correspond to M, in which case encode selects one such seed uniformly at random. (Every seed, however, corresponds to a unique message).We require that encode be efficiently invertible. In other words, given S, we can decode through the inverse DTE decode(S) =M, which returns S's unique corresponding message. With a DTE that gives strong security (as we explain later), decode accurately generates pm. In this case, selecting S uniformly at random from {0, 1}ℓ and decoding to obtain M = decode(S) returns approximately the original pm. In other words, the DTE is a good model of the message distribution.

2.2 Another View of HE

Honey encryption introduces many interesting technical challenges. A good DTE is tailored to the message distribution over which encryption is taking place. Constructing a DTE for highly structured data, such as credit card numbers, is relatively straightforward but can otherwise be challenging. For example, constructing a DTE for the databases in password managers requires understanding how users select suites of passwords. Currently, researchers only have a good understanding of user selection of individual passwords (thanks to the Rock You breach). Fortunately, DTEs don't have to be perfect to provide useful security. HE raises other interesting challenges. For instance, what happens if a user mistypes a password and decrypts a cipher text (for example, an encrypted password database) into a wrong but valid looking message? Several approaches to this typo-safety problem exist, such as associating different images or colors with different plaintexts, as in checking decrypted passwords online automatically. Identifying the best approach remains an interesting open problem.

B. Distribution servers

Cloud computing data security is an evolving sub domain of network security, computer security and information security. Data is stored on different servers. So the clients have to trust on cloud service provider on the data availability as well as data security [15]. Sometimes cloud service provider may hide data integrity and data corruptions issues from clients to maintain the reputation, to avoid this problem we introduce flexible and an effective third party auditing mechanism. Third party auditor is like an inspector. Third party auditor used for audits the user out sourced data. Third party auditor helps data owner to make sure that his information is safe in cloud [12]. Cloud users save his data in cloud servers so that data reliability as well as data security is primary concern. Data integrity verification at untrusted server is one of the biggest concerns with cloud data storage [2] [3].

C. Adversary Model

With cloud storage security threats may occur in different ways come from two different sources. First one, cloud service provider can be self-interested, envious and possibly disloyal. Not only its desire to move data but it may also attempt to hide a data loss incident due to management errors, convoluted failures and so on. Second one, it may also exist an economically motivated attack, who has the capability to compromise a number of cloud data storage servers at various time intervals and

consequently able to delete or modify users data while resting undetermined by the cloud service provider [12]. Clearly, there are two types of adversary with different levels of capability. The adversary which is interested in corrupting the user's data files stored on individual servers is called weak adversary. Once a server is comprised, weak adversary can pollute the original data files by modifying or introducing its own fraudulent data to prevent the original data from being retrieved by the user. The adversary can intentionally modify the data files as long as they are internally consistent by compromising all the storage servers called strong adversary. It is same to the case wither all servers are colluding together to hide a data loss or corruption incident [6].

IV. RELATED WORK TO ACHIEVE SOLUTIONS IN DATA SECURITY

To the best of our knowledge, this is the first work that addresses the problem of securing data stored in multiple server systems when the cryptographic material is exposed. We survey relevant related work in the areas of all-or-nothing transformations, bastion: security against key exposure secret-sharing techniques.

A. All or Nothing Transformations

All-or-nothing transformations (AONTs) are not encryption, but frequently make use of symmetric ciphers and may be applied before encryption. In precise terms, "an AONT is an unkeyed, invertible, randomized, complex way of transformation, with the property that it is difficult to invert unless all of the output is known. The maximum of AONTs leverage a secret key that is embedded in the output blocks. Once all output blocks are available, the key can be recovered and single blocks can be inverted. Therefore AONT is not an encryption mechanism and does not need the decryptor to have any key material. Resch et al. [5] combine AONT and information dispersal to provide both fault-tolerance and data secrecy, in the context of distributed storage systems.

B. Bastion: Security against Key Exposure

Bastion, which ensures that plaintext data cannot be recovered as long as the adversary has access to all but two cipher text blocks—even when the encryption key is exposed. Bastion first encrypts the data with one round of block cipher encryption, and then applies an efficient linear post-processing to the cipher text. By doing so, Bastion relaxes the notion of all-or-nothing encryption at the benefit of increased performance. More specifically, the first round of Bastion consists of CTR mode encryption with a randomly chosen key K, i.e., $y' = Enc(K, x)$. The output cipher text y' is then fed to a linear transform[8]. Namely, our transform basically computes $y = y' \cdot A$ where A is a square matrix such that: (i) all diagonal elements are set to 0, and (ii) the remaining off-diagonal elements are set to 1.

C. Secret Sharing

Secret sharing mechanism [5] allows a dealer to share a secret among a number of shareholders, such that only authorized subsets of shareholders can renovate the secret. In threshold secret sharing mechanisms [11,7], the dealer defines a threshold t and each set of shareholders of cardinality equal to or greater than t is authorized to renovate the secret. Secret sharing insure security against an un-authorized subset of shareholders; however, they incur a high computation or storage cost, which makes them impractical for sharing large data's. Rabin [4] put forward an information dispersal algorithm with smaller overhead than the one of [7], however the proposal in [4] does not provide any security guarantees when a small number of shares (less than the reconstruction threshold) are available. Krawczyk[14] proposed to combine both Shamir's [7] and Rabin's[4] approaches; in [14] a file is first encrypted using AES and then dispersed using the scheme in [4], while the encryption key is shared using the scheme in [7]. In Krawczyk's scheme,

individual cipher text blocks encrypted with AES can be decrypted once the key is exposed.

V. RESULTS

Figure 2.1 shows the login page for data owner and data users. It also contains auditor details for updating all information. Data owner can authorize through admin where the data creator is used to upload files. Newest user is accessed through user details.

Figure 2.1. Login Page

Once the data owner authorized, he/she able to upload any kind of file in the application that shows in fig 2.2

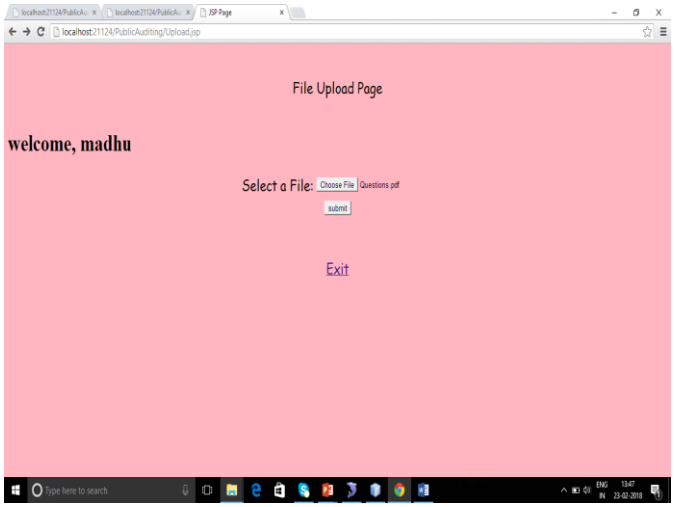

Figure 2.2. Files Upload Page

Create account at https://console.aws.amazon.com and login into AWS. The next step is to select elastic beanstalk from the domain. In the application, select the environment like PHP, Java, tomcat etc., then deploy the selected file in AWS shows in fig 2.3

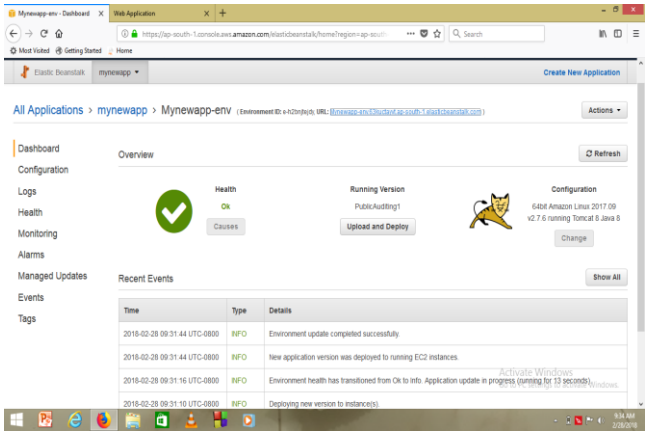

Figure 2.3. Aws Elastic Beanstalk

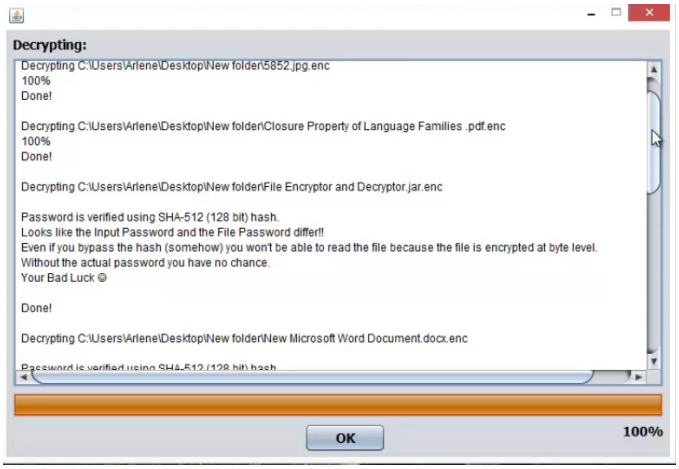

Figure 2.4. Decryption Of File

VI. CONCLUSION

A secure and efficient privacy preserving public auditing scheme is proposed. Privacy preserving public auditing is achieved with the help of TPA. It does the auditing without retrieving the data copy, therefore preserving the privacy of the data. The data has split into parts and then stored in the encrypted format at cloud for storage, thus maintaining the confidentiality of data. The cloud server is used only to store the encrypted form of data. Cloud server is not involved in verification proof computing, thus reduces online burden of cloud server. The proposed method satisfies all the auditing requirements as well as it reduces cloud server burden.

VII. REFERENCES

[1]. Ghassan O. Karame, Claudio Soriente,"Securing Cloud Data under Key Exposure", IEEE Transactions on Cloud Computing, 2017.

[2]. Swapnali S. More and Sangita S. Chaudhari, "Secure and efficient public auditing scheme for cloud storage" ,2016.

[3]. Ahmed Albugmi, Madini O. Alassafi, Robert Walters and Gary Wills ,"Data security in cloud computing" 2016.

[4]. T. Subha, S.Rabin and S. Jayashri, "Efficient privacy preserving integrity checking model for cloud data storage security", 2016.

[5]. P.Mell and T.Grance, "The NIST of Cloud Computing", NIST Special Publication 2011.

[6]. C. Wang, Q. Wang, K. Ren, and W. Lou, " Ensuring Data Storage Security in Cloud Computing" IWQoS'09,Charleston, South Carolina, USA, 2009.

[7]. Yuan Zhang, Lei Niu, Guomin Yang, Yong. Shamir Yu, Yi Mu, "On the security of auditing mechanisms for secure cloud storage",2014.

[8]. Guomin Yang and Lai Yan-Ming, "Cryptanalysis of a simple key for access control based on polynomial" 2013.

[9]. Li H, Dai Y, Tian L, Yang H, "Identity-based authentication for cloud computing,(LNCS)2009.

[10]. Yuan Zhang, Chunxiang Xu, Jining Zhao, Xiaojun Zhang, Lei Niu, Guomin Yang, "Cryptanalysis of an Integrity checking scheme for cloud data sharing" , Journal of Information Security and applications 2015.

[11]. N. Penchalaiah and R. Seshadri, "Effective Comparison and evaluation of DES and Rijndael Algorithm (AES)," International Journal of Computer Science and Engineering, 2010.

[12]. Mahima Joshi and Yudhveer Singh Moudgil. "Secure Cloud Storage. International Journal of Computer Science & Communication Networks", Vol. 1 (2), pp. 171-175, 2011.

[13]. S Dhanaya. "Privacy preserving third party auditing in cloud. Dissertation" report, 2015.

[14]. W. Zeng, Y Zhao, K. Ou, and W. Song. "Research on Cloud Storage Architecture and Key Technologies", 2009.

[15]. S. More and S. Chaudhari, "Third party public auditing scheme for cloud storage," Procedia Computer Science, 2016.