Retraction

Retraction: Efficient Method for Storing Health Record in Cloud Using Integrity Auditing and Data Sharing (J. Phys.: Conf. Ser. 1916 012191)

Published 23 February 2022

This article (and all articles in the proceedings volume relating to the same conference) has been retracted by IOP Publishing following an extensive investigation in line with the COPE guidelines. This investigation has uncovered evidence of systematic manipulation of the publication process and considerable citation manipulation.

IOP Publishing respectfully requests that readers consider all work within this volume potentially unreliable, as the volume has not been through a credible peer review process.

IOP Publishing regrets that our usual quality checks did not identify these issues before publication, and have since put additional measures in place to try to prevent these issues from reoccurring. IOP Publishing wishes to credit anonymous whistleblowers and the Problematic Paper Screener [1] for bringing some of the above issues to our attention, prompting us to investigate further.

[1] Cabanac G, Labbé C and Magazinov A 2021 arXiv:2107.06751v1

Retraction published: 23 February 2022
Efficient Method for Storing Health Record in Cloud Using Integrity Auditing and Data Sharing

K Rama Abirami\textsuperscript{1}, Ajaye K P\textsuperscript{1}, Amrrish Roshan B L\textsuperscript{1}, Arun Nishanthan A\textsuperscript{1}
\textsuperscript{1}Department of CSE, Sri Krishna College of Engineering and Technology, Coimbatore
ramaabirami@skcet.ac.in

Abstract. Identity based cloud storage authorization is proposed in this paper to provide protection of cloud data. Many sensitive information is stored in the cloud, and it must be kept safe from unauthorized access. By hiding the sensitive information’s, privacy for an individual or organization is provided. The data in the cloud are secured by hiding the sensitive information. In the proposed system when the user and data analyzer sign-in they will receive the verification mail to assure as a valid user or data analyzer. The verification key for the user and the data analyzer is generated by the Private Key Generator (PKG). The user and the data analyzer enter the cloud using the verification key to upload and retrieve data from the cloud. Until uploading data to the cloud, the user encrypts the data, concealing confidential details. The Third Party Auditor is in charge of the cloud auditing process (TPA). This scheme allows data files stored in the cloud to be shared and used by data analyzers under the condition that confidential information is hidden, allowing the auditing system to operate more efficiently.

Keywords: Cloud storage; Data integrity auditing; Data sharing; Sensitive information hiding

1. Introduction

Cloud computing provides a variety of computing services such as servers, storage, databases, networking, applications, and information over the Internet, or “the cloud,” in order to provide quicker innovation, more adaptable tools, and economies of scale. It better protects our data by uploading it on a large scale over the Internet to an offsite cloud storage system that is accessible from any location and computer. Many governments, administrations, and individuals choose to store data in the cloud. Most cloud providers provide a comprehensive collection of terms, technologies, and controls to help us improve our overall security posture and protect our data, applications, and infrastructure from potential threats. However, due to unavoidable virus and hardware burdens, data stored in the cloud can be tampered with or compromised. The Integrity Auditing scheme is proposed in order to ensure that data stored in the cloud is safe or that data stored in the cloud is available\textsuperscript{[14]}. When a user have a file and he want to upload them to cloud, then the user first need to login and he want be an authenticated user. After getting authentication only user can upload the data. Data is generally distributed through multiple users in cloud storage applications like Google Drive, Dropbox, and iCloud. One of the most common features of cloud storage is data sharing, which enable a user to share their information. Some of the shared data in the cloud could contain confidential information. Electronic Health Records (EHRs) stored and exchanged in
the cloud, for example, typically contain sensitive information about patients (such as their name, phone number, and ID number) as well as sensitive information about hospitals (such as their name) [9]. If EHRs are directly submitted to the cloud and exchanged for future testing purposes, confidential patient and hospital information will invariably be revealed to the cloud and researchers. Furthermore, due to the presence of human errors and software/hardware failures within the cloud, the confidentiality of the EHRs must be ensured. As a result, it's critical to complete remote data integrity auditing on some condition that allows the data to be shared while protecting confidential data.

2. Similar Work

A variety of remote data integrity auditing systems have been proposed to verify the integrity of data stored in the cloud[12]. A Third Party Auditor (TPA) is used to ensure the validity of cloud reporting needs on behalf of the client and to reduce the user's computational burden. Provable Data Possession (PDP) was first suggested by [1][10] to secure data handling on a not known cloud. Non probability methods and public key authentication tokens are used in their proposed scheme to achieve blockless verification and lower I/O costs. Paper [2] proposed a practical scheme and described a method called Proof of Retrievability (PoR). The information stored in the cloud is frequently retrieved in this scheme, and therefore the data's integrity is frequently assured. [7] design an improved PoR scheme with full proofs of security in the security model. [3] suggested a remote system security auditing method that preserves confidential information by using a random camouflaging technique. [4] used a unique random masking technique to create a far-flung data integrity auditing scheme that protects data privacy. When compared to the scheme in [3], the data integrity auditing scheme is more effective. To minimise the computational complexity of approaches to understanding on the user side, [5] devised a remote data integrity auditing scheme based on the indistinguishability difficult technique. [6] built a simple remote data quality compliance frameworkal using a Third Party Medium (TMP). The TMP assists users in creating signatures under the condition that data privacy is also secured.

Due to its ability to provide a scalable, dynamic, and resilient platform for both educational and commercial environments, cloud computing has recently risen in popularity as a viable alternative to conventional computing[13]. In a public cloud environment, the client moves their data to a public cloud server (PCS), and they have no control over their remote data. As a consequence, information protection, such as data confidentiality, integrity, and availability, is a critical issue in cloud storage. Under certain circumstances, the client is unable to access the remote data in his or her hands, such as when the client is incarcerated for a felony, on a marine ship, on the battleground during a battle, and so on. It wants to act as a stand-in for the remote data possession checking task. [8] investigate proxy provable data ownership in this paper (PPDP). When the client is unable to conduct the remote data possession testing in public clouds, PPDP can be critical. The PPDP framework model, security model, and style process are all investigated[11]. We created a well-organized PPDP protocol based on the bilinear pairing process. Our protocol has been proven to be stable and efficient through security and performance testing.

The aim of this research is to develop advanced encryption standards (AES) that are better suited for controlling access to data stored in the cloud. For this resolution, we focus on giving encrypt or complete control over access rights, including in the case of multiple independent authorities, and allowing viable user revocation. It also provides adequate key management, which is critical in practise[15]. This scheme is made possible by deleting the work load of a termination event from the cloud service provider in exchange for a small, but acceptable, permanent overhead of the encryption and decryption algorithms that the users would travel through. As a consequence, rather than putting the computation overhead on one party (such as a private network), which could quickly escalate to a
performance slowdown, the computation overhead is distributed over a potentially large number of users. Our scheme's formal security proof is based on the odd add and standard cosine group models.

Advance Encryption Standards (AES) with contracting out decryption allows for fine-grained encryption data sharing. It overcomes the limitations of standard AES schemes, such as ciphertext size and decryption expense. For example, an ABE scheme with decryption enables a third party, such as a cloud server, to read the data, to change an ABE ciphertext to use a public transformation key given by the user, create an El Gamal-style ciphertext, allowing the latter to be decrypted more easily than the former. The original outsourced ABE scheme has a drawback in that the quality of the data is not checked by the customer. In other words, an customer may be persuaded to accept a faulty or intentionally transformed output. In this document, we first introduce a verification key within the performance of the encryption algorithm to formalise an AES security model with verifiable outsourced decryption. Then we show how to transform any AES scheme with external provider decryption into an AES scheme with testable external provider decryption using a process. The new method is simple, universal, and nearly perfect. When checking the similarities, the original external provider ABE to our proposed AES, except for some anti operations like key algorithms, it has no effect on the user’s or cloud server's computation costs. It also does not extend the ciphertext size, but it does add a hash value that is less than 20 bytes for an 80-bit security level. We have an in-depth performance assessment to illustrate the advantages of our strategy, as well as a concrete construction that supports Green et al.'s ciphertext-policy's AES scheme with outsourced decryption.

The chosen encryption technology for solving the difficult problem of secure data sharing in cloud computing is ciphertext-policy attribute-based encryption (CP-ABE). N. R. Borkar, Sphurti Atram, and others Multilevel hierarchy is a common feature of shared data files, particularly in the field of healthcare and, by extension, the military. In CP-ABE, the shared file hierarchy architecture has not found. For cloud storage, this paper proposes an efficient file structure indicator cryptographic algorithm. The layered access structures have been combined into one. The transformed access structure is then used to encrypt the hierarchical files. The files will share the attributes associated with the ciphertext components. As a consequence, both ciphertext storage and encryption time are reduced. Furthermore, under the quality statement, the proposed scheme is proven to be stable. In terms of encryption and decryption, the proposed scheme is very successful, as demonstrated by experimental simulation. When the amount of the files increases with the benefits of our scheme it becomes more and noticeable.

Tianyu Zhao et al. suggest ABE is a public key cryptography scheme that enables users to encrypt and decrypt data using user attributes. In this document, we look at how to create an advanced encryption standards (AES), when the qualities supplier is also the holder of the communications to be encrypted and exchanged scheme is used. In comparison to other ABE schemes, our scheme decrypts relatively quickly in this setting, resulting in a small ciphertext. Our AES concept is also very compatible with communications that have already been encrypted when our system is built, so encrypted data can be used directly in our framework without having to be re-encrypted. We equate our scheme's efficiency to Bethencourt's in this article.

3. Problem Identification

To secure personal data in remote system security auditing, a new concept called personality shared data security auditing with personal data hiding for safe cloud storage has been proposed. This scheme would protect confidential information while allowing other information to be transmitted. User must get verified and user want to be an authenticated user to upload the data. The private key of the user ID will be verified by the ID provided by the user. If user ID wants to retrieve the file, it may give the user a message. The data slabs that refer to the file's confidential details are refined using a refiner. Later, the
refined file is checked and stored in cloud storage, where the confidential information is protected from unauthorised access. Three packages (User, Authenticator, Cloud Server, and Third Party Auditing) were used in this project. Until uploading data to the cloud, the user must first be checked by an authenticator and generate ID for data slabs, according to this remote data integrity auditing scheme. During the integrity auditing process, these IDs are used to ensure that the cloud actually owns these data slabs. The user then uploads these data slabs to the cloud, along with their corresponding ID. Data stored in the cloud is commonly shared by multiple users in cloud storage applications like Google Drive, Dropbox, and I Cloud. One of the most popular features of cloud storage is data sharing, which enables multiple users to share their data with one another. However, some confidential information is stored in this shared data in the cloud.

Patients' personal details (patient's name, telephone number, and ID number) will usually be stored and exchanged in cloud-based Electronic Health Records (EHRs). Until uploading data slabs to the cloud, the user must first be authenticated and create an ID for them. During the integrity auditing process, this ID is used to prove that the cloud actually holds these data block. The data owner then uploads these information slabs to the cloud, along with their corresponding ID. Many cloud storage apps, such as Google Drive, Dropbox, and I Cloud, enable multiple users to share data stored in the cloud. One of the most popular features of cloud storage is data sharing, which enables multiple users to share their data with one another. However, some confidential information is stored in this shared data in the cloud.

Patients' sensitive data (patient's name, phone number, and ID number, for example) is stored and exchanged in the cloud Electronic Health Records (EHRs). If these EHRs are directly uploaded to the cloud and exchanged for research purposes, it would cause a slew of issues, including the direct exposure of confidential data owner information to the cloud and researching student. Furthermore, due to the fact of human mistake and software/hardware mistake in the cloud, the credibility of EHRs must be guaranteed. As a result, it's critical to complete remote data integrity auditing under the condition that shared data's confidential information is protected. One possible solution is to encrypt the entire shared file before uploading it to the cloud, create the ID that will be used to verify the encrypted file's integrity, and then upload the encrypted file and its corresponding ID to the cloud. Since only the user can decrypt this file, this method can hide sensitive information. However, it will prevent anyone from accessing the entire shared file. Encrypting the EHRs of infectious disease patients, for example, will preserve and improve the safety of data owner, but these encrypted EHRs can no longer be used by researching student. A possible solution to the above problem is to give the researchers the decryption key. This approach is infeasible to accept in a real-world environment for the following reasons. To begin with, distributing decryption keys necessitates safe networks, which can be difficult to achieve in certain cases. Furthermore, it appears to be extremely difficult for a consumer to predict which researchers will use his or her EHRs in the near future when the EHRs are uploaded to the cloud. As a result, encrypting the entire shared file to mask confidential information is impossible. Data sharing is necessary and useful when confidential information is concealed in remote data integrity auditing.

### 3.1. Disadvantage

- It requires multi-cloud storage system.
- It does not hide sensitive information.
- It just stores the health record but it’s not secure.
- It stores the data directly without any verification process.

### 4. Proposed System
In the proposed system, we are going to guard and encrypt the sensitive information by the Advance Encryption Standards (AES) techniques. When the user login in and before share the file to cloud the user want to get authenticated. The files that are stored in the clouds are grounded on their attributes that is included in the authentication permission offered by the Authenticator. Authenticator offer the authentication permission to the user by sending mail and verify the user for uploading data in the cloud. The user blinds the data slabs that refer to the file's personal sensitive information before creating the corresponding ID. These IDs are used to ensure that the file is authenticated and that its integrity is maintained. The user then saves this blinded file in the cloud and sends the refiner its corresponding ID.

These shielded data blocks, as well as the data blocks corresponding to the organization's secret data, are refined by the distributor after receiving the message from the customer, and then changes the ID of refinement data blocks into correct ones for the refinement file. The distributor uploads the refined file and its associated ID to the server. In the process of integrity auditing, these signatures are used to check the integrity of the refined file. If the TPA needs to check the credibility of the refined file stored in the cloud, it will submit auditing challenges to the cloud. The cloud then offers an auditing evidence of data tenure to the TPA. The TPA verifies the refined file's accuracy by confirming whether or not the auditing proof is right. The sensitive information in this auditing will be blinded TPA can't view the sensitive information. The proposed model will perform efficiently and is feasible. An architecture diagram is given (figure 1), we can refer that architecture diagram for the clear explanation.

4.1. Proposed Architecture Modelling

4.2. Implementation

The theoretical aspect of the project is brought to the working stage during implementation. As a result, it can be regarded as a crucial stage in completing a successful new system and gaining consumer confidence in its proper operation. Safe preparation, review of previous work and its limitations on implementation, and design of methods to achieve changeover are all part of the implementation stage.

4.2.1. Secret key Generation

4.2.1.1. Consumer Module
In this consumer module is take input as secure parameter k. The master secret key msk and the system public parameters pp are the outputs. PKG’s Extract (pp;msk; ID) method is used to extract data. All inputs include the system public parameters pp, the master secret key msk, and a user's identity ID. After obtaining the user's identity ID, the PKG selects a value at random and computes the user ID's private key. The user's private key, skID, is returned. The user can only accept skID as his private key if it passes the accuracy validation. The signature was then generated with the original file F, the user’s secret key skID, the user's signing secret key ssk, and the holder name as inputs. The user ID produces a verification key by choosing a value at unique. The user ID then selects a seed as the input secret key of the pseudo-random function f at random. The hidden seed is used by the consumer ID to measure the hiding factor, which is used to hide the data blocks containing personal sensitive data. The user ID must blind the data blocks corresponding to the original file's F private personal data before sending it to the distributor to protect the personal sensitive information from the distributor. These data blocks indexes are complete. The consumer ID computes the hidden data blocks m of the hidden file F, the signature I on block m, and the file tag calculation. In the refinement algorithm, the user ID calculates a transformation value that is used to convert the signature. Using the blinding factor, the user ID will recover the original file F. It produces a blinded file F, as well as a signature collection and a file tag for it.

4.2.1.2. Authentication
The authentication permission was provided by Authenticator and that verify the user and approve user as an authenticated user. Based on their files attribute, Verifier grants the user consent to upload a file. The use of advanced cryptographic standards is carried out. The data can be downloaded from the cloud using the term.

4.2.1.3. Private data sanitization
The distributor checks the ID to see if the file tag is correct. If the ID is correct, the refiner will tell you how to get the name of the file identifier and the verification values. The refiner refines the blinded data slabs that relate to the confidential details of the company. The user ID has blinded the data slabs whose indexes are in set 1, causing the contents of these data slabs to become messy code. Wildcards may be used by the distributor to overwrite the contents of these data blocks. F’ is sent to the cloud by the distributor, who then sends the file tag to the TPA. Finally, the messages are removed from device.

4.2.1.4. Security Computing
The TPA checks the file tag's validity. The TPA will not conduct computing tasks if the file tag is not correct. The TPA then investigates how to acquire the file identifier name and authentication values, and creates an auditing challenge as a result. The cloud generates a proof of data ownership after collecting auditing challenge data from the TPA. The TPA verifies the auditing proof's accuracy.

5. Result
The following figures show the result of our implementation.
Figure 2. home page

The figure 2 shows the home page of the application from where user can select the options.

Figure 3. login page

This figure 3 shows the login page for the data owner.

Figure 4. Data upload page
The above figure 4 shows the page where the data owner upload the health records with the mentioned details.

![Figure 5. user login page](image)

The figure 5 shows the data user login.

![Figure 6. search documents page](image)

The figure 6 shows the search page from where the data user can search the required medical records.

![Figure 7. owner details page](image)
The figure 7 shows the data owner's details.

Figure 8. User details page

The figure 8 shows the details of data user.

6. Conclusion

An identity-based data integrity auditing scheme for safe cloud storage that facilitates data sharing while hiding confidential information was proposed in this paper. The Authenticator provides authentication for the user in our proposed scheme, and once the user is validated as an authenticated user, the user is allowed to upload and download files from the cloud. The proposed scheme achieves sufficient protection and performance, according to the security proof and experimental review.

References

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, Provable data possession at untrusted stores, in Proceedings of the 14th ACM Conference on Computer and Communications Security, ser. CCS ’07, 2007, pp. 598–609.
[2] Juels and B. S. Kaliski, Pors: Proofs of retrievability for large files, in Proceedings of the 14th ACM Conference on Computer and Communications Security, ser. CCS ’07, 2007, pp. 584–597.
[3] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, “Privacy-preserving public auditing for secure cloud storage,” IEEE Transactions on Computers, 62(2), pp. 362–375, 2013.
[4] S. G. Worku, C. Xu, J. Zhao, and X. He, Secure and efficient privacy-preserving public auditing scheme for cloud storage, Comput. Electr. Eng., 40(5), pp.1703–1713, Jul. 2014.
[5] C. Guan, K. Ren, F. Zhang, F. Kerschbaum, and J. Yu, Symmetric-key based proofs of retrievability supporting public verification, in Computer Security – ESORICS 2015. Cham: Springer International Publishing, 2015, pp. 203–223.
[6] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, and R. Hao, Light-weight and privacy-preserving secure cloud auditing scheme for group users via the third party medium, Journal of Network and Computer Applications, 82, pp. 56–64, 2017.
[7] H. Shacham and B. Waters, Compact proofs of retrievability, J. Cryptology, 26(3), pp. 442–483, Jul. 2013.
[8] H. Wang, Proxy provable data possession in public clouds, IEEE Transactions on Services Computing, 6(4), pp. 551–559, 2013.
[9] J. Sun and Y. Fang, Cross-domain data sharing in distributed electronic health record systems, *IEEE Transactions on Parallel and Distributed Systems*, 21(6), pp. 754–764, June 2010.

[10] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, Scalable and efficient provable data possession, *in Proceedings of the 4th international conference on Security and privacy in communication netowrks*, 2008, pp. 1–10.

[11] C. Erway, C. Papamanthou, and R. Tamassia, Dynamic provable data possession, *in ACM Conference on Computer and Communications Security*, 2009, pp. 213–222.

[12] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, Enabling public auditability and data dynamics for storage security in cloud computing, *IEEE Transactions on Parallel and Distributed Systems*, 22(5), pp. 847–859, May 2011.

[13] D. Devikanniga, A. Ramu, and A. Haldorai, Efficient Diagnosis of Liver Disease using Support Vector Machine Optimized with Crows Search Algorithm, *EAI Endorsed Transactions on Energy Web*, p. 164177, Jul. 2018. doi:10.4108/eai.13-7-2018.164177.

[14] H. Anandakumar and K. Umamaheswari, Supervised machine learning techniques in cognitive radio networks during cooperative spectrum handovers, *Cluster Computing*, vol. 20, no. 2, pp. 1505–1515, Mar. 2017.

[15] J. Yu and H. Wang, Strong key-exposure resilient auditing for secure cloud storage, *IEEE Transactions on Information Forensics and Security*, 12(8), pp. 1931–1940, Aug 2017.