FACILITATING IDENTITY-BASED INTEGRITY AUDITING AND DATA SHARING WITH SENSITIVE INFORMATION HIDING FOR CLOUD STORAGE SECURITY

RADHIKA RK

Dept. of Computer Applications, SNGIST, Ernakulum, India

---***---

Abstract - *With cloud storage services, users can maintain and manage their data and make that data accessible over a network usually the internet. Remote data integrity auditing is proposed to assure the integrity of data stored in the cloud. In cloud storage system say the Electronic Health Record system, cloud file will have some sensitive information which should not be uncover to others when it is shared. By encrypting the whole shared file it will realize the sensitive information hiding. But the problem is that it will make this shared file unable to be used by others. Data sharing with sensitive information hiding has been still not explored. Here we propose a remote data integrity auditing scheme which make data sharing possible with sensitive information hiding. Sanitizer is used to sanitize data block that is corresponding to the sensitive information of the file. It also transforms data block signatures into valid ones for the sanitized file. In the phase of integrity auditing these signatures are used to verify integrity of sanitized file. On that account, our proposed scheme which is based on identity-based cryptography makes the file stored in the cloud to be shared and used by others with sensitive information hiding.*

Key Words: **Data integrity auditing, Sanitizer, Cloud storage, Encrypting, Sensitive information**

1. INTRODUCTION

With the rapid growth of data it has become difficult for users to store the data locally. Thus users would like to store their data in the cloud. But the data stored in the cloud might be lost or corrupted because of unavoidable human errors, bugs in software and hardware problems. Remote data integrity auditing scheme is proposed to ensure that whether the data is stored correctly in cloud.

Before the data block is uploaded to the cloud data owner needs to generate signatures for them in remote data integrity schemes. This generated signatures are used to prove that the cloud have these data blocks in the phase of integrity auditing. Data owner upload data block along with their corresponding signatures to the cloud. Data sharing which is a common feature of cloud storage allows users to share their data with others. Shared data in the cloud will have some sensitive information. For example, the Electronic Health Records stored and shared in the cloud contain patient's sensitive information (patient's name, telephone number etc.) and hospital's sensitive information (hospital's name, etc.). For research purposes if these EHRs are uploaded to the cloud, sensitive information of patient and hospital will be uncover to the researchers. In cloud since software, hardware failures can happen, the integrity of EHRs need to be undertaken. Thus it is essential to carry out remote data integrity auditing so that the sensitive information of shared data is protected.

A possible procedure of solving this problem is to encrypt the shared file before sending it to the cloud after that generate signatures, which is used to verify the integrity of this encrypted file. Lastly upload this encrypted file and its corresponding signatures to the cloud. This method can recognize the sensitive information hiding as only the data owner can decrypt this file. But, this will make whole shared file unable to be used by others. For instance, if we encrypt the EHRs of infectious disease patients it can protect the privacy of patient and hospital. But these encrypted EHRs cannot be beneficially utilized by researchers. Issuing the decryption key to the researchers can be a solution to the problem mentioned above. Yet it is absurd to adopt this method in real scenarios due to the following reasons. Firstly, distributing decryption key needs secure channels, which is hard to be satisfied in a few instances. Besides, it seems very challenging for a user to know which researchers will use their EHRs in the near future when user uploads the EHRs to the cloud. Because of that it is impractical to hide sensitive information by encrypting the whole shared file. Therefore how to realize data sharing with sensitive information hiding in remote data integrity is very important.

2. LITERATURE ISTING SYSTEM

K Ren, C. Wang, and Q. Wang [1], As cloud computing became more powerful in IT industry, more problems are reported by academics and practitioners. In this paper our objective is to attain an understanding of types of issues and challenges that have been emerging over past five years and identify gaps between the focus of the literature and what practitioners deem. A systematic literature review as well as interviews with experts has been conducted to answer our research questions. Our findings suggest that researchers have been mainly focusing on issues related to security and privacy, infrastructure and data management.

Interoperability across different service providers has also been an active area of research. Despite the significant overlap between the topics being discussed in the literature and the issues raised by the practitioners, our findings show that some issues and challenges that practitioners consider important are understudied.

G. Ateniese, R. Bums, R. Curtmola, L. Kissner, Z. Peterson and D. Song [2] In this paper author define the ID-based RDIC and its security model. It also includes the security against cloud server and privacy against a third party verifier.ID-based RDIC protocol does not leak the information of stored data at the time of verification of RDIC process. The result shows that it's secure against server in the generic group model. The proposed model is very secure.

A. Juels, B. S. Kaliski [3] In this paper author define proofs of retrievability. A POR strategy authorizes back-up service to produce compact proof that a user can recover a target file F. A POR may be observed as a type of cryptographic proof of knowledge which is designed to handle a large file F. We inspect POR protocols here in which the communication costs, storage requirements of the user are small parameters that are essentially independent of the length of file F. In addition to proposing practical POR constructions, author investigates implementation considerations and optimizations that bear on previously explored, related schemes. PORs give rise to a new and unusual security definition whose formulation is another contribution of our work.

H. Shacham, and B. Waters [4] In a proof-of-retrievability system, a data storage center persuade a verifier that he is actually storing all client's data. The main challenge is to build system that is both efficient and secure. Client's data should be extracted from any prover that passes a verification check. In this paper, we give the first proof-of-retrievability schemes with full proofs of security against arbitrary adversaries in the strongest model, that of juels and Kaliski. First scheme, built from BLS signatures and secure in the random oracle model, has the shortest query and response of any proof-of-retrievability with public verifiability. Second scheme, which builds elegantly on PRFs and is secure in the standard model, has the shortest response of any proof-of-retrievability scheme with private verifiability.

C. Wang, S. S. M. Chow, Q. Wang, K. Ren and W. Lou[5] Enabling public auditability for cloud storage is important so that user can resort to a third-party auditor to check the integrity of outsourced data and be worry free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy and to introduce no additional online burden to user.

S. G. Worku, C. Xu, J. Zhao, and X. He [6] To regenerate the data at corrupted servers in the cloud storage in absence of data owners, we propose public auditing scheme. Public auditing scheme includes TPA and a semi Trusted Proxy Server to improve the integrity of data and to regenerate the data. To check the integrity of data TPA does the periodical verification of servers in the cloud. If the faulty server is found it sends it to the proxy for regenerate. The semi trusted proxy server also makes the data owners free from online burden. A homomorphic novel authenticator which uses the BLS signature is designed which is more appropriate for regenerating the data at corrupted servers. Privacy preserving is also allowed by using the encryption mechanism.

C. Guan, K. Ren, F. Zhang, F. Kerchbaumn and J. YU [7] In this work we explore in distinguishability obfuscation for building a Proof-of-Retrievability scheme that provides public verification while the encryption is based on symmetric key primitives. The resulting scheme offers light-weight storing and proving at the expense of longer verification. This could be useful in applications where outsourcing files is usually done by low-power client and verifications can be done by well-equipped machines. We also show that the proposed scheme can support dynamic updates. For better assessing our proposed scheme, we give a performance analysis of our scheme and a comparison with several other existing schemes which demonstrates that our scheme achieves better performance on the data owner side and the server side.

3. EXISTING SYSTEM

With the enormous amount of data it is burden on users to store data locally. Many organizations and individuals wish to store data on the cloud. The data that is stored on cloud can be lost or corrupted due to software, hardware problems and human errors. Cloud storage does bring out few security threats to data owners. Because of some serious security problems many cloud users would not like to use cloud storage. A fundamental concern of cloud users is the integrity of their outsourced files. There are few things that might lead to data corruption. Firstly, cloud service providers are not fully trusted. Thus for financial reason the cloud service provider might delete the data that are unique or have not been accessed so that it can save the space for storing other files for charging extra expenses. Secondly, the stored data could be corrupted due to cloud server failure or adversary attacks. Yet in order to maintain a good reputation cloud service provider may knowingly hide data loss events. In cloud storage primary concern of cloud users of cloud users are data integrity and leakage.

3.1 Example for Existing System

Fig-1 shows pictorious description for EHRs. EHRs contain data with sensitive information in two parts. First is personal sensitive information such as patient's name, id, etc. Second is organizational sensitive information such as hospital name, etc. Sanitizer can be regarded as administrator of the HER information system in a hospital. Personal sensitive information should not be exposed to sanitizer. It should also be not exposed to the cloud and the shared users. EHR of patient's data are generated by doctor which is send to sanitizer for storing in EHR information system. But these EHRs normally contain sensitive information of patient and hospital.

To retain the privacy of patient from the sanitizer, doctor will blind the patient's sensitive information of each EHR before sending it to sanitizer. It is doctor who generate signature for blinded EHR and send it to sanitizer. These messages are stored into EHR information system by sanitizer. When doctor needs the EHR, he sends request to the sanitizer. From EHR information system sanitizer downloads the blinded EHR and send them to doctor. At last, the doctor recovers the original HER from this blinded EHR. On the point of uploading and sharing this EHR to cloud for research purpose, the sanitizer needs to sanitize the data blocks corresponding to the patient's sensitive information of the EHR. To retain the privacy of hospital, sanitizer needs to sanitize the data blocks corresponding to the hospital's sensitive information. Wildcards are used to replace data blocks. Sanitizer can transform data blocks signatures into valid ones for sanitized EHR.

Sanitizer does not need to interact with doctors. At last, sanitized EHRs and their corresponding signatures are uploaded to the cloud by sanitizer. By this method, EHR can be shared and used by researchers with sensitive information of EHRs hidden. Sanitizer is essential because of below mentioned reasons. The contents of data blocks may become messy code after the data blocks corresponding to sensitive information are blinded. By using wildcard to replace the contents of the data block, sanitizer can unify the format. Sanitizer sanitizes data blocks corresponding to sensitive information which protects the privacy of the hospital. If doctor needs the EHR, sanitizer as the administrator of EHR information system can download the blinded EHR and send it to doctor, who recover original HER from blinded EHR. Sanitizer can sanitize the EHRs in bulk and upload it to the cloud at a fixed time.

3.2 Disadvantages of Existing System

For achieving data sharing with sensitive information hiding, we think making use of the idea in the sanitizable signatures to sanitize the sensitive information of the file by introducing an authorized sanitizer. However, it is infeasible if these sanitizable signatures are directly used in remote data integrity auditing. First, the signature used is based on PKI, which suffers from complicated certificate management. Second, the signature used does not support block less verifiability. And last this signature is constructed based on chameleon hashes. But lots of chameleon hashes shows the key exposure problem. To avoid this security problem, signature used requires strongly unforgeable chameleon hashes, which will cause huge computation overhead.

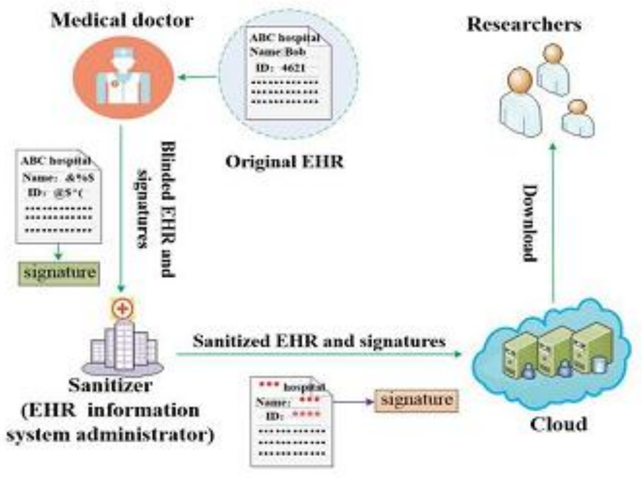

Fig -1: Example of EHRs

4. PROPOSED SYSTEM

We design a new efficient signature algorithm in the phase of signature generation that addresses the drawback of existing system. It supports block less verifiability that allows the verifier to check the integrity of data without downloading entire data from the cloud. It is based on identity-based cryptography, which simplifies the complicated certificate management.

Here the PKG generate the private key for user according to their identity. The correctness of the received private key can be checked by user. When user want to upload data to the cloud, user needs to use a blinding factor to blind data blocks which is corresponding to personal sensitive information of the original file. If necessary using blinding factor, user can recover original file. To generate signatures for blinded file user can use the designed signature algorithm. User will generate a file tag that is used to assure the correctness of the file identifier name and some verification values. The user also figures out a transformation value that is used to transform signatures for sanitizer. Lastly, the user sends the file tag along with the transformation value and the blinded file with its corresponding signatures to the sanitizer. Just as the above message from user is valid, sanitizer first sanitizes the blinded data blocks into uniform format. It also sanitizes the data blocks corresponding to the hospital's sensitive information to protect the privacy. It also transforms their corresponding signatures into valid one for sanitized file by using transformation value. At last, the sanitizer uploads the sanitized file and the corresponding signatures to the cloud.

When the data integrity auditing task is completed, the cloud develops an auditing proof according to the challenge from the TPA. The TPA can check integrity of sanitized file stored in the cloud. This is done by checking whether auditing proof is correct or not. In addition to these, the researcher and doctor can post review and can even rate the review. This is achieved through sentimental analysis using natural language processing.

4.1 Advantages of Proposed System

I. In this system the sensitive information is hidden with help of Double Encryption.
II. In this System the file stored in the cloud can be shared and used by others given the condition that the sensitive information is protected using a secret key.
III. The concept of block level is used to randomly store data and maintain security.
IV. Both security and privacy of the patient's records is maintained.

5. SYSTEM DESIGN

The system models have five kinds of different entities. They are:

I. Cloud: The cloud provides excessive data storage space to the user. Users can upload their data with others through the cloud storage services.
II. User: The user is a member of an organization that has a large number of files to be stored in the cloud.
III. Sanitizer: The sanitizer sanitizes the data blocks corresponding to the sensitive information which include personal and organization's sensitive information in the file. It transforms these data block's signature into valid one for the sanitized file. It also uploads the sanitized file and its corresponding signature to the cloud.
IV. Private Key Generator: The Private Key Generator is responsible for generating system public parameters and the private key for the user according to his identity ID.
V. Third Party Auditor: The Third Party Auditor is a public verifier. It verifies the integrity of data stored in the cloud on behalf of users.
PKG generates private keys for user and proxy depending upon their identities. Cloud storage server permit user to outsource its data to the cloud. But user is bothered about the security of their data. Therefore it authorizes TPA to do data auditing task routinely to make sure that its data is stored correctly.

5.1 Identity-Based Auditing Scheme

An identity-based auditing scheme consists of seven algorithms: Setup, Extract, PSKGen, TagGen, Challenge, Proof, and Verify.
I. Setup (1^k) → (params, mpk, msk): This algorithm accepts k which is security parameter as input and outputs system public parameter params, master public key mpk and master secret key.
II. Extract (mpk, msk, params, ID) → sk_{ID} : This algorithm accepts the system public parameters pp, the master secret key msk, and a user's identity ID as input. The user's private key sk_{ID} is obtained as output
III. PSKGen (params, ID_u, ID_p) → u: This algorithm accepts system parameters params, identity of user ID_u, identity od proxy ID_p as input. Proxy signature key u is obtained as output.
IV. TagGen (F, u) → σ: This algorithm accepts file F and proxy signature u as input. It computes and outputs the corresponding tag of data blocks $\sigma = (\sigma_1, \sigma_{2,\dots\dots}, \sigma_n)$.
V. Challenge (F_{info}) → C: This algorithm accepts the information of data file *F* as input and outputs the challenge set *C*.
VI. Proof (F, C, σ) → P: This algorithm accepts data file F, challenge set C, from TPA and verification tags σ as input and outputs a response proof P
VII. Verify (C,P, params, mpk, F_{info}) → 0/1: This algorithm accepts challenge set C, response proof P, system parameters params, master public key mpk, information of data file F_{info} as input and outputs the auditing result 0 or 1.

5.2 Goals of Design

To accurately support data sharing with sensitive information hiding in identity-based integrity for secure cloud storage, our scheme is designed to achieve the following goals.

I. Correctness: It includes Private Key correctness, the correctness of the blinded file and its signatures and auditing correctness. Private Key correctness is to make sure that when PKG sends a correct private key to the user, private key can pass the verification of user. The correctness of the blinded file and its corresponding signatures is to assure that when the user sends a blinded file and its corresponding signatures to the sanitizer, the blinded file and its corresponding signatures generated by user can pass the verification of the sanitizer. Auditing correctness is to make sure that when the cloud properly stores the user's sanitized data, the proof it generates can pass the verification of the TPA.

II. Sensitive information hiding: For ensuring that sensitive information of the file is not exposed to the sanitizer and also to ensure all of the sensitive information of the file is not exposed to the cloud and the shared users.

III. Auditing soundness: This is to ensure that if the cloud does not store user's intact sanitized data it cannot pass the TPA's verification.

5.3 System Architecture

Fig-2 shows pictorious description for system architecture.The user firstly blinds the data blocks corresponding to sensitive information in the file and then generates corresponding signatures. The signatures are used to verify integrity of the file and to guarantee the authenticity of the file. After that user send blinded file and its corresponding signatures to the sanitizer. When sanitizer receive message from user, it sanitizes the blinded data blocks and the data blocks corresponding to the sensitive information. It also transforms the signatures of sanitized data blocks into valid ones for the sanitized file. Finally, the sanitized file and its corresponding signatures are send to cloud by sanitizer. In the phase of integrity auditing, signatures are used to verify integrity of sanitized file. When the TPA wants to verify integrity of the sanitized file which is stored in cloud, it sends an auditing challenge to the cloud. Cloud responds to TPA by an auditing proof of data possession.

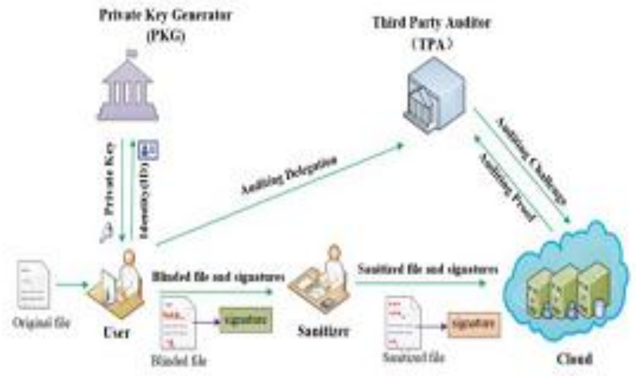

Fig -2: System Architecture

5.4 Working

There are two user sections, doctor and researcher. The doctor uploads data with seed value. Also the doctor and the researcher can search for data using the seed. But the researcher could only see the needed data and doctor can view whole data. Behind this while uploading data the PKG creates the private key for user as per their identity ID. Then the information blocks are blinded using the blind factor which is derived from the seed value. After that signatures are created for the blinded file. These signatures are utilized to check the integrity of this blind file. Additionally, the file tag created is employed to make sure the correctness of the file identifier name and a few verification values. Sanitizer uses the transformation values to make signatures for blind file. Finally, using the blind file, its corresponding signatures, and also the file tag along with the transformation worth the sanitizer sanitize the file. After sanitization the file send to the cloud. It is TPA which checks whether the cloud truly possess the file.

6. CONCLUSION

In this paper, the problem of secure communication is solved. Here we propose an identity based integrity auditing and sensitive information hiding in data sharing scheme for cloud storage security. It supports information sharing with sensitive information hiding and setting of review for the system. In this scheme, the file kept in the cloud are shared and utilized by others on the condition that the sensitive information of the file is protected. Besides, the remote information integrity auditing is possible. This work effectively stores and retrieves the records from the cloud space database server. The records are encrypted and decrypted whenever necessary so that they are secure.

This work is credible with lots of potential and nuances for development. Some of them are listed below:

I. The application if developed as web services, then many applications can make use of the records.
II. The data integrity in cloud environment is not considered. The error situation can be recovered if there is any mismatch.
III. The web site and database can be hosted in real cloud place during the implementation.

REFERENCES

[1] K Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," IEEE Internet Computing, vol. 16, no.1, pp. 69-73, Jan 2012.
[2] G. Ateniese, R. Bums, R. Curtmola, 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.
[3] A. Juels, 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. 594-597.
[4] H. Shacham, and B. Waters, "Compact proofs of retrieva bility," J. Cryptology, vol.26, no.3, pp. 442-483, Jul 2013.
[5] 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, vol.62, no.2, pp. 362-375.
[6] S. G. Worku, C. Xu, J. Zhao, and X. He, "Secure and efficient Privacy-preserving public auditing scheme for cloud Storage," Comput. Electr. Eng., vol. 40, no. 5, pp. 1703-1713, Jul. 2014.
[7] C. Guan, K. Ren, F. Zhang, F. Kerchbaumn and J. YU, "Symmetric-key based proofs of retrievability supporting public verification," in Computer Security – ESORICS 2015, pp. 203-223.