Journal of Engineering Sciences
Vol 11, Issue 12, DEC/2020
ISSN NO:0377-9254

EFFICIENT DATA AUDITING SCHEME USING FUZZY PRIVATE KEY IN CLOUD STORAGE SYSTEMS

[1]NUKAM REDDY SRINADH, [2]A. SHOBITHA LAKSHMI

[1]Assoc. Professor, Dept. of CSE, Visvodaya Engineering College, Kavali, Andhra Pradesh, India.

[2]PG Scholar, Dept. of CSE, Visvodaya Engineering College, Kavali, Andhra Pradesh, India.

Abstract – To ensure the integrity of the data stored in the cloud, many data integrity auditing schemes have been proposed. In most, if not all, of the existing schemes, a user needs to employ his private key to generate the data authenticators for realizing the data integrity auditing. Thus, the user has to possess a hardware token to store his private key and memorize a password to activate this private key. If this hardware token is lost or this password is forgotten, most of the current data integrity auditing schemes would be unable to work. In order to overcome this problem, we propose a new paradigm called data integrity auditing without private key storage and design such a scheme. In this scheme, we use biometric data as the user's fuzzy private key to avoid using the hardware token. Meanwhile, the scheme can still effectively complete the data integrity auditing. We utilize a linear sketch with coding and error correction processes to confirm the identity of the user. In addition, we design a new signature scheme which not only supports blockless verifiability, but also is compatible with the linear sketch.

Index terms – Cloud Computing, Blockless Verifiability, fuzzy biometric data.

I. INTROUDUCTION

Cloud storage can provide powerful and on-demand data storage services for users. By using the cloud service, users can outsource their data to the cloud without wasting substantial maintenance expenditure of hardware and the users upload their data to the cloud, they will lose the physical control of their data since they no longer keep their data in local. Thus, the integrity of the cloud data is hard to be guaranteed, due to the inevitable hardware/software failures and human errors in the cloud.

Many data integrity auditing schemes have been proposed to allow either the data owner or the Third Party Auditor (TPA) to check whether the data stored in the cloud is intact or not. These schemes focus on different aspects of data integrity auditing, such as

data dynamic operation, the privacy protection of data and user identities, key exposure resilience, the simplification of certificate management and privacy-preserving authenticators, etc.

In the above data integrity auditing schemes, the user needs to generate authenticators for data blocks with his private key. It means that the user has to store and manage his private key in a secure manner. In general, the user needs a portable secure hardware token (e.g. USB token, smart card) to store his private key and memorizes a password that is used to activate this private key. The user might need to remember multiple passwords for different secure applications in practical scenarios, which is not user friendly. In addition, the hardware token that contains the private key might be lost.

Once the password is forgotten or the hardware token is lost, the user would no longer be able to generate the authenticator for any new data block. The data integrity auditing will not be functioning as usual. Therefore, it is very interesting and appealing to find a method to realize data integrity auditing without storing the private key.

A feasible method is to use biometric data, such as fingerprint and iris scan, as the private key. Biometric data, as a part of human body, can uniquely link the individual and the private key. Unfortunately, biometric data is measured with inevitable noise each time and cannot be reproduced precisely since some factors can affect the change of biometric data. For example, the finger of each person will generate a different fingerprint image every time due to pressure, moisture, presentation angle, dirt, different sensors, and so on. Therefore, the biometric data cannot be used directly as the private key to generate authenticators in data integrity auditing.

II. BACKGROUND WORK

Ateniese et al. firstly proposed the notion of Provable Data Possession (PDP). They employed the random sample technique and homomorphic linear authenticators to design a PDP scheme, which allows an auditor to verify the integrity of cloud data without downloading the whole data from the cloud. Juels and Kaliski proposed the concept of Proof of Retrievability (PoR). In the proposed scheme, the errorcorrecting codes and the spot-checking technique are utilized to ensure the retrievability and the integrity of the data stored in the cloud. Shacham and Waters constructed two PoR schemes with private verifiability and public verifiability by using pseudorandom function and BLS signature.

To support user-interactions, including data modification, insertion and deletion, Zhu et

al. constructed a dynamic data integrity auditing scheme by exploiting the index hash tables. Sookhak et al. also considered the problem of data dynamics in data integrity auditing and designed a data integrity auditing scheme supporting data dynamic operations based on the Divide and Conquer Table. In public data integrity auditing, the TPA might derive the contents of user's data by challenging the same data blocks multiple times. To protect the data privacy, Wang et al. exploited the random masking technique to construct the first public data integrity auditing scheme supporting privacy preserving. Li et al. proposed a data integrity auditing scheme which preserves data privacy from the TPA. Yu et al. proposed a cloud storage auditing scheme with perfect data privacy preserving by making use of zero-knowledge proof. To relieve the user's computation burden of authenticator generation, Guan et al. constructed a data integrity auditing scheme using indistinguishability obfuscation technique, which reduces the overhead for generating data authenticators.

Li et al. proposed a data integrity auditing scheme which contains a cloud storage server and a cloud audit server. In this scheme, the cloud audit server helps user to generate data authenticators before uploading data to the cloud storage server. Shen et al. designed a light-weight data integrity auditing scheme, which introduced a Third Party Medium to generate authenticators and verify data integrity on behalf of users.

The data sharing is used widely in cloud storage scenarios. To protect the identity privacy of user, Wang et al. proposed a shared data integrity auditing scheme based on the ring signature. Yang et al. designed a remote data integrity auditing scheme for shared data, which supports both the identity privacy and the identity traceability. By using the homomorphic verifiable group signature, Fu et al. proposed a privacy-aware remote data integrity auditing scheme for shared data. In order to achieve efficient user revocation, Wang et al. designed a shared data integrity auditing scheme supporting user revocation by making use of the proxy re-signature. Based on the identity-based setting, Zhang et al. constructed a cloud storage auditing scheme for shared data supporting real efficient user revocation. To realize the data sharing with sensitive information hiding, Shen et al. designed an identity-based cloud storage auditing scheme for shared data.

Other aspects, such as eliminating certificate management and key exposure resilience in data integrity auditing have also been studied. However, all of existing remote data integrity auditing schemes do not take the

problem of private key storage into account. In this paper, we explore how to achieve data integrity auditing scheme without private key storage for secure cloud storage.

III. PROPOSED WORK

As illustrated in Fig. 1, the system model involves three types of entities: the user, the cloud, and the TPA. The cloud provides enormous data storage space to the user. The user has a large number of files to be uploaded to the cloud. The TPA is a public verifier who is delegated by the user to verify the integrity of the data stored in the cloud.

In the phase of user registration, the biometric data (e.g. fingerprint) is extracted from the user who wants to use the cloud storage service. When a data owner would like to upload data to the cloud, he firstly extracts biometric data as his fuzzy private key and randomly generates a signing key. Then, this data owner computes authenticators for data blocks with his signing key. Finally, he uploads these data blocks along with the authenticator set to the cloud and deletes these messages from the local storage. In the phase of data integrity auditing, the TPA verifies whether the cloud truly keeps the user's intact data or not by executing the challenge-response protocol with the cloud.

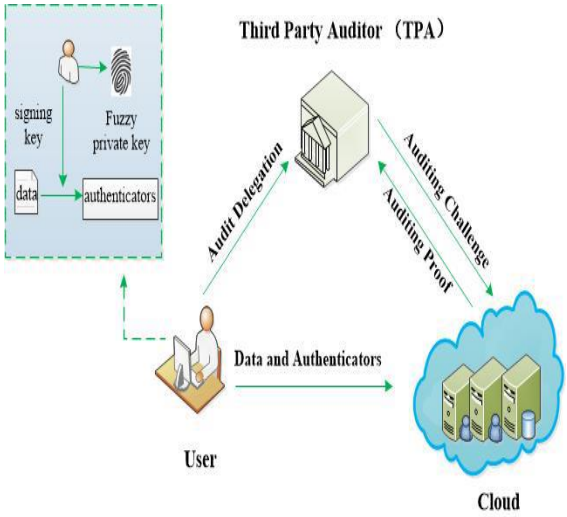

Fig. 1: System Overview

Implementation Algorithm

A data integrity auditing scheme without private key storage consists of the following five algorithms: Setup, KeyGen, Sign- Gen, ProofGen and ProofVerify. Specifically, these algorithms are described as follows:

1) Setup(1^k, *FKS*): This algorithm takes as input a fuzzy key setting *FKS* and a security parameter *k*. It outputs the public parameter *pp'*.

2) KeyGen(*pp', y*): This algorithm takes as input the public parameter *pp'* and the biometric data y ∈ Rn. It generates *pk* as his public key, which including a sketch c and a verification key *vk*.

3) SignGen(*y', F*) This algorithm takes as input the biometric data y' ∈ Rn and the file *F*. It outputs a signature which includes the verification key *vk'*, the sketch *c'* and the set of authenticators Φ.

4) ProofGen(F, Φ, *chal*) This algorithm takes as input the file F, the corresponding authenticator set Φ and the auditing challenge *chal*. It outputs an auditing proof P that proves the cloud indeed keeps this file.

5) ProofVerify(pk, *chal*, P, vk', c') This algorithm takes as input the user's public key pk, the auditing challenge *chal*, the auditing proof P, the verification key vk' and the sketch c'. The TPA verifies the correctness of proof P.

IV. RESULT ANALYSIS

In this section, we evaluate the performance of our proposed scheme in experiments. We run these experiments on a windows machine with an Intel Pentium 2.70GHz processor and 4GB memory. Our scheme is implemented by utilizing C programming language with the GNU Multiple Precision Arithmetic (GMP) and the free AES-128 scheme. We set the base field size to be 512 bits, the size of an element in Z_p^* to be 160 bits and the size of a shared cloud file to be 20MB.

1) Computation overhead

a) Authenticator generation. In order to evaluate the efficiency of authentication generation of our scheme, we compute the authenticators for different blocks from 0 to 1000 increased by an interval of 100. Fig. 2 shows that the computation overhead of authenticator generation linearly increases with the number of data blocks. The running time varies from 1.5s to 12.9s.

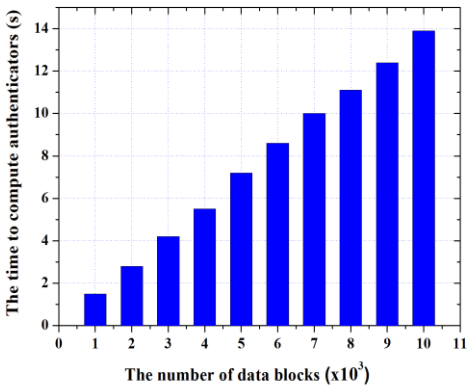

Fig. 2. The computation overhead of authenticator generation

b) Auditing. In order to evaluate the performance of auditing in our scheme, we respectively show the time spent on the TPA and the cloud. The experimental results are presented in Fig. 3 and Fig. 4. In the experiment, we choose to challenge different blocks from 0 to 1000 increased by an interval of 100. From Fig. 3, we have the observation that the auditing computation overhead of the TPA is mainly from challenge generation and proof verification. The running time of challenge generation ranges from 0.038s to 0.395s. The running time of proof verification is linear with the number of the challenged data blocks, ranging from 0.795s to 8.685s. As shown in Fig. 4, the running time of proof generation ranges from 0.401s to 3.793s on the cloud side. From the above experiments, we can infer that the auditing computation overhead

of the TPA and the cloud both linearly increases with the number of the challenged blocks. The trade-off here is that, with more challenged blocks, the result of integrity auditing is more accurate, meanwhile, the auditing work gets more cumbersome.

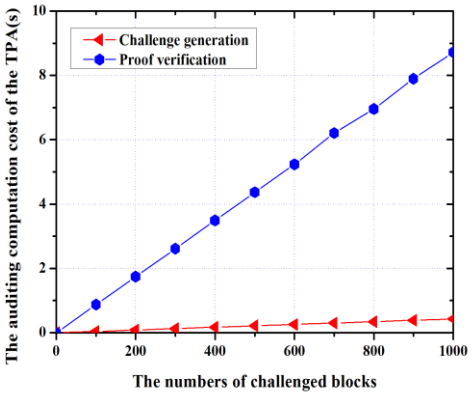

Fig. 3. The computation overhead of the TPA in the phase of auditing

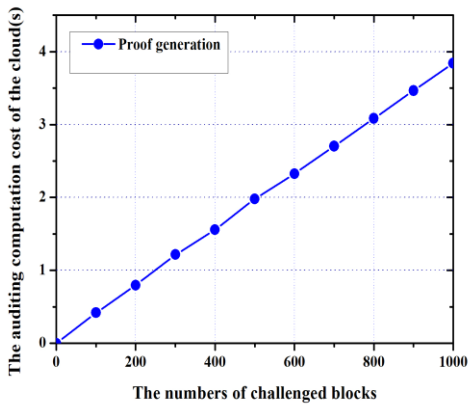

Fig. 4. The computation overhead of the cloud in the phase of auditing

2) Communication overhead

We evaluate the communication overhead of the auditing phase in our scheme. As discussed previously, the communication overhead is mainly from the challenge overhead and proof overhead. The challenge $chal = \{i, \beta_i\}_{i \in I}$ costs $c.(|s| + |p|)$, which has a linear relationship with the number c of the challenged blocks. The proof $P = \{\mu, \sigma, vk', c'\}$ costs $|p| + 2|q| + |W|$, which is independent from the number c of the challenged blocks. From Fig. 5, we can see that the communication overhead of challenge message linearly increases with the number of the challenged blocks, while the communication overhead of proof message is constant.

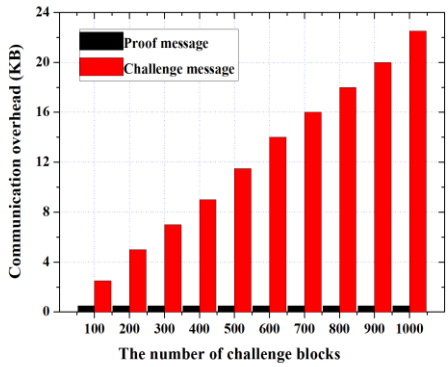

Fig. 5. The communication overhead of challenge message and proof message

V. CONCLUSION

In this paper, we concentrated how to execute bio-metric to acknowledge information honesty reviewing without putting away private key. We propose the principal pragmatic information honesty inspecting plan without private key

stockpiling for secure distributed storage. In the proposed conspire, we use biometric information as client's fluffy private key to accomplish information honesty inspecting without private key stockpiling. The conventional security confirmation and the presentation examination show that our proposed plot is provably secure and effective.

REFERENCES

[1] H. Dewan and R. C. Hansdah, "A survey of cloud storage facilities," in 2011 IEEE World Congress on Services, July 2011, pp. 224–231.

[2] 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.

[3] A. F. Barsoum and M. A. Hasan, "Provable multicopy dynamic data possession in cloud computing systems," IEEE Transactions on Information Forensics and Security, vol. 10, no. 3, pp. 485–497, March 2015.

[4] N. Garg and S. Bawa, "Rits-mht: Relative indexed and time stamped merkle hash tree based data auditing protocol for cloud computing," Journal of Network & Computer Applications, vol. 84, pp. 1–13, 2017.

[5] H. Jin, H. Jiang, and K. Zhou, "Dynamic and public auditing with fair arbitration for cloud data," IEEE Transactions on Cloud Computing, vol. 13, no. 9, pp. 1–14, 2014.

[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] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," in International Conference on Applied Cryptography and Network Security, 2012, pp. 507–525.

[8] B. Wang, H. Li, and M. Li, "Privacy-preserving public auditing for shared cloud data supporting group dynamics," in 2013 IEEE International Conference on Communications (ICC), June 2013, pp. 1946–1950.

AUTHORS

Mr. NUKAM REDDY SRINADH received his B.Sc Degree in Mathematics, Physics and Chemistry from Sri Venkateswara University, Tirupti, A.P , India in 1997, Master of Computer Applications from Sri Venkateswara University in 2000,Master Of

Technology in computer science Engineering From AAIDU in 2006. Now He is pursuing Ph.D. from Rayalaseema University, Kurnool, AndhraPradesh, India. His research areas include Computer Networks/Secure Routing in MANET.

Ms. A. SHOBITHA LAKSHMI has received her MCA degree at S.V.U PG Center affiliated to SV University in 2011 and pursuing M.Tech degree in Computer Science and Engineering at Visvodaya Engineering College affiliated to JNTUA in 2018-2020.