International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 17
November 2016

Basic Solutions for the Key-Exposure Problem of Cloud Storage Auditing

[1]SHRUTIRANI KHANDRE.A, [2]adavelli Ramesh, [2]machukonda kishore

[1]SLC's College of Engineering and Technologies, Hyderabad
[2]Associate Professor, CSE Department,
[3]Assitent Professor, CSE Department,

Abstract: **Auditing is an important service to verify the data in the cloud. Most of the auditing protocols are based on the assumption that the client's secret key for auditing is secure. The security is not fully achieved, because of the low security parameters of the client. If the auditing protocol is not secured means the data of the client will exposed inevitably. In this paper a new mechanism of cloud auditing is implemented. And investigate to reduce the damage of the client key exposure in cloud storage auditing. Here the designing is built upon to overcome the week key auditing process. The auditing protocol is designed with the help of key exposure resilience. In the proposed design, the binary tree structure and the pre-order traversal technique is used to update the secret keys of the client. The security proof and the performance shows the cloud storage auditing with key exposure resilience is very efficient.**

Keywords: **Cloud Storage Auditing, Client Key Exposure**

I. INTRODUCTION

Cloud computing is a computing paradigm, where a large pool of systems are connected in private or public networks, to provide dynamically scalable infrastructure for application, data and file storage. The large amount of data is stored in the cloud. To verify the integrity of a data which is stored on the cloud, the cloud storage auditing is used. Auditing is an integrity check in the cloud data base. It is an important checking in the cloud auditing protocols that are highly researched on recent years. Each protocols act as a different auditing mechanism. The aim of introducing the protocol is to achieve high bandwidth and computation efficiency. Thus in this project Homomorphic Linear Authenticator (HLA) is used for an Efficient auditing scheme.

The efficiency of the (HLA) technique is, it supports block less verification. It is used to reduce the overheads of computation and communication auditing. The auditor is used to verify the integrity of the data in cloud without retrieving the whole data.

The privacy protection of data is an important aspect of cloud storage auditing. It is used to reduce the computational burden of the client. The third party auditor is introduced to help the client to periodically check the integrity of data in cloud. Auditing protocols are for the privacy of data in cloud.

II. RELATED WORK

Some existing remote integrity checking methods can only serve for static archive data and thus cannot be applied to the auditing service since the data in the cloud can be dynamically updated. Thus, an efficient and secure dynamic auditing protocol is desired to convince data owners that the data are correctly stored in the cloud. In [1] paper, the design of an auditing framework for cloud storage systems and propose an efficient and privacy-preserving auditing protocol. Then, we extend our auditing protocol to support the data dynamic operations, which is efficient and provably secure in the random oracle model. We further extend our auditing protocol to support batch auditing for both multiple owners and multiple clouds, without using any trusted organizer.
The analysis and simulation results show that our proposed auditing protocols are secure and efficient, especially it reduce the computation cost of the auditor. Provable data possession (PDP) is a technique for ensuring the integrity of data in storage outsourcing. In this paper, we address the construction of an efficient PDP scheme for distributed cloud storage to support the scalability of service and data migration, in which we consider the existence of multiple cloud service providers to cooperatively store and maintain the clients' data. We present a cooperative PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy [2]. And [3] in this paper the mechanism of storage auditing is also proposed.

A model for provable data possession (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs [5].

III. PROBLEM STATEMENT

The Key exposure resilience in the storage auditing protocol is not fully supported in the existing system

this mechanism is used to detect any dishonest, such as deleting or modifying some client's data that is stored in the cloud in previous time periods can all be detected, even if the cloud gets the clients current secret key for cloud storage auditing.

Auditing protocols can also support dynamic data operations. Other aspects, such as proxy auditing, user revocation and eliminating certificate management in cloud storage auditing have also been studied. Though many research works about cloud storage auditing have been done in recent years, a critical security problem exposure problem for cloud storage auditing, has remained unexplored in previous researches. While all existing protocols focus on the faults or dishonesty of the cloud, they have overlooked the possible weak sense of security and/or low security settings at the client.

Unfortunately, previous auditing protocols did not consider this critical issue, and any exposure of the client's secret auditing key would make most of the existing auditing protocols unable to work correctly. We focus on how to reduce the damage of the client's key exposure in cloud storage auditing. Our goal is to design a cloud storage auditing protocol with built-in key-exposure resilience. How to do it efficiently under this new problem setting brings in many new challenges to be addressed below.

First of all, applying the traditional solution of key revocation to cloud storage auditing is not practical. This is because, whenever the client's secret key for auditing is exposed, the client needs to produce a new pair of public key and secret key and regenerate the authenticators for the client's data previously stored in cloud. The process involves the downloading of whole data from the cloud, producing new authenticators, and re-uploading everything back to the cloud, all of which can be tedious and cumbersome.

Besides, it cannot always guarantee that the cloud provides real data when the client regenerates new authenticators. Secondly, directly adopting standard key-evolving technique is also not suitable for the new problem setting. It can lead to retrieving all of the actual files blocks when the verification is preceded. This is partly because the technique is incompatible with block less verification. The resulting authenticators cannot be aggregated, leading to unacceptably high computation and communication cost for the storage auditing.

IV. PROPOSED METHOD

In this paper two basic solutions for the key-exposure problem of cloud storage auditing is discussed and implemented. The first is a naive solution, which in fact cannot fundamentally solve this problem. The second is a slightly better solution, which can solve this problem but has a large overhead. They are both impractical when applied in realistic settings. And then we give our core protocol that is much more efficient than both of the basic solutions.

A. Naive Solution:

In this solution, the client still uses the traditional key revocation method. Once the client knows his secret key for cloud storage auditing is exposed, he will revoke this secret key and the corresponding public key. Meanwhile, he generates one new pair of secret key and public key, and publishes the new public key by the certificate update. The authenticators of

the data previously stored in cloud, however, all need to be updated because the old secret key is no longer secure. Thus, the client needs to download all his previously stored data from the cloud, produce new authenticators for them using the new secret key, and then upload these new authenticators to the cloud. Obviously, it is a complex procedure, and consumes a lot of time and resource. Furthermore, because the cloud has known the original secret key for cloud storage auditing, it may have already changed the data blocks and the corresponding authenticators. It would become very difficult for the client to even ensure the correctness of downloaded data and the authenticators from the cloud. Therefore, simply renewing secret key and public key cannot fundamentally solve this problem in full.

B. Slightly Better solution:

The client initially generates a series of public keys and secret keys: (PK 1, SK 1), (PK 2, SK 2) · · · (PK). Let the fixed public key be (PK 1 · · · PK T) and the secret key in time period j be (SK j · · · SK). If the client uploads files to the cloud in time period j, the client uses SK T to compute authenticators for these files. Then the client uploads files and authenticators to the cloud. When auditing these files, the client uses PK to verify whether the authenticators for these files are indeed generated through SK j. When the time period changes from j to j + 1, the client deletes SK from his storage. Then the new secret key is (SK j j+1, SK T· · · SK This solution is clearly better than the naive solution.

The Naive and Slightly Better solutions are impractical when applied in realistic settings. Cloud Storage Auditing with Key-exposure Resilience protocol is used in proposed system which is much more efficient than both of the basic solutions.

C. Proposed Model:

The proposed model is described in below Fig 1. In this process the entire model of this paper is explained here.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 17
November 2016

In this model the Cloud Storage Auditing with Key-exposure Resilience protocol is used. The key update algorithm helps to update the secrete key for each time period.

Key Update in each
time period
Challenge
Cloud
Auditing
Proof
Client
Data Flow
Security Message flow

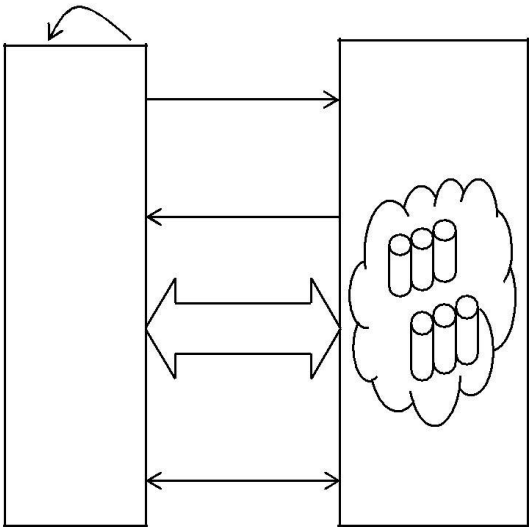


Fig 1 Proposed Model

Our goal is to design a practical auditing protocol with key- exposure resilience, in which the operational complexities of key size, computation overhead and communication overhead should be at most sub linear to T.

In order to achieve our goal, we use a binary tree structure to appoint time periods and associate periods with tree nodes by the pre-order traversal technique.

The secret key in each time period is organized as a stack. In each time period, the secret key is updated by a forward - secure technique. It guarantees that any authenticator generated in one time period cannot be computed from the secret keys for any other time period later than this one. Besides, it helps to ensure that the complexities of keys size, computation overhead and communication overhead are only logarithmic in total number of time periods T.

As a result, the auditing protocol achieves key-exposure resilience while satisfying our efficiency requirements. As we will show later, in our protocol, the client can audit the integrity of the cloud data still in aggregated manner, i.e., without retrieving the entire data from the cloud. As same as the key-evolving mechanisms, our proposed protocol does not consider the key exposure resistance during one time period. Below, we will give the detailed description of our core protocol.

The cloud auditing protocol with key exposure resilience protocol helps to protect the data from the unauthorized user. It helps to verify the integrity of the data.

The auditing protocol with key-exposure Resilience:

An auditing protocol with key-exposure resilience is composed by five algorithms (SysSetup, KeyUpdate, AuthGen, ProofGen, ProofVerify) shown below.

1. SysSetup:

It is the first algorithm that is first setup the input parameter k and the total time period T. here the parameters that used in this algorithms is K and T. and finally it will generate an output as an public key PK. This was generated by the client.

2. KeyUpdate:

It is a probabilistic algorithm. It will take the input as public key pk. For denoting the current period where the data to be position is find out by the parameter j. For the first period the current data that is denoted by the client secret key is SK_j. And the next time period the current time is denoted as SK_{j+1}. This algorithm is also run by the client side.

3. AuthGen:

It is also termed as Authentication generated Algorithm. This algorithm is used to authenticate the file that should be used for process. This algorithm is also generated in client side.

4.* ProofGen*:

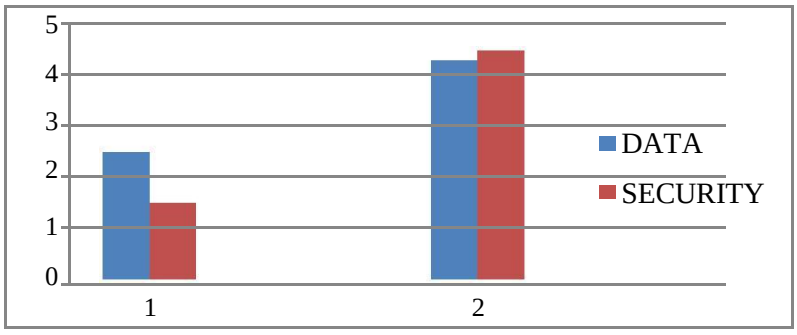
This algorithm is used to verify the sign value of the system. This value is issued by the auditor. This algorithm is generated by the cloud side.

5.* ProofVerify*:

Proof verification is done by the client side was the proof should be used to find the required authority or not.

V. PERFORMANCE EVALUATION

The performance Evaluation of the auditing frame work in this paper is expressed below. In this paper the data that stored in the cloud and the security ratio is denoted. In this process both the value is compared with the proposed concept. By using the key update algorithm, the security of the data is high.



Thus by comparing to that aspects proposed model ensure the data security in cloud comparing to the existing system the ratio is high.

VI. CONCLUSION

As mentioned before data security in cloud is not efficient and the key exposure problem is a big problem when there is any third party auditing done in the cloud. This can be overcome by achieving the best binary tree structure and the pre-order traversal technique. This can be further implemented by the proof of verifiability by the Auditor. The methods that used to bind with each other will increase the efficiency and performance of the proposed model.

VII. FUTURE ENHANCEMENT

The cloud storage auditing with key exposure resilience protocol is used in paper .The user can upload their data in the cloud and they can protect their data by using the Third Party Auditor. The key update algorithm is used to protect the client's key from the unauthorized user. In paper, the data owner independently upload the data to the Cloud and it is difficult to monitor the data and checking the process in offline. Thus data owner stands in online for integrity checking. This can be achieved by introducing Proxy component to check for the integrity. This is an added advantage to the data owner that he need not stay online for integrity checking. The data owner provides a key to the proxy server using that key proxy is responsible for checking the data. This should be considering as the future work to overcome this drawback.

REFERENCES

[1] G. Ateniese et al., "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Comput. Commun. Secur. 2007, pp. 598–609.

[2] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. 4th Int. Conf. Secur. Privacy Commun. Netw. 2008, Art. ID 9.

[3] F. SEbE, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. Knowl. Data Eng., vol. 20, no. 8, pp. 1034– 1038, Aug. 2008.

[4] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple replica provable data possession," in Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst., Jun. 2008, pp. 411–420.

[5] H. Shacham and B. Waters, "Compact proofs of Retrievability," in Advances in Cryptology ASIACRYPT. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.

[6] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," IEEE Netw., vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.

[7] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in
Proc. 17th ACM Conf. Comput. Commun. Secur., 2010, pp. 756–758.

[8] K. Yang and X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities,"
World Wide Web, vol. 15, no. 4, pp. 409–428, 2012.

[9] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 9, pp. 1717–1726, Sep. 2013.

[10] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy preserving public auditing for secure cloud storage," IEEE Trans. Comput., vol. 62, no. 2, pp. 362–375, Feb. 2013.

[11] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847–859, May 2011.

[12] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for

outsourced storages in clouds," IEEE Trans. Services Comput., vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.

[13] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. 16th ACM Conf. Comput. Commun. Secur. 2009, pp. 213–222.

[14] H. Wang, "Proxy provable data possession in public clouds," IEEE Trans. Services Comput., vol. 6, no. 4, pp. 551– 559, Oct./Dec. 2013.

[15] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in Proc. IEEE INFOCOM, Apr. 2013, pp. 2904–2912.

[16] H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based remote data possession checking in public clouds,"
IET Inf. Secur., vol. 8, no. 2, pp. 114–121, Mar. 2014.

[17] T. Stewart. (Aug. 2012). Security Policy and Key Management: Centrally Manage Encryption Key. [Online].
Available: http://www.slideshare.net/Tina-stewart/security-policy-and-enterprise-key-management-fromvormetric

[18] Microsoft. (2014). Key Management. [Online]. Available: http://technet.microsoft.com/en-us/library/cc961626.aspx

[19] FBI. (2012). Is Your Computer Infected with DNSChanger Malware? [Online]. Available: http://www.fbi.gov/news/ news_blog/is-yourcomputer- infected-with-DNSChanger-malware

[20] FBI. (2011). Botnet Operation Disabled. [Online]. Available:
http://www.fbi.gov/news/stories/2011/april/botnet_041411