IJIEMR Transactions, online available on 31st Oct 2017. Link :http://www.ijiemr.org/downloads.php?vol=Volume-6&issue=ISSUE-9

Title: ENABLING CLOUD STORAGE WITH KEY EXPOSURE

Volume 06, Issue 09, Pages: 430–435.

Paper Authors

S.ARSHIYA SULTHANA, P.NAMRATHA
Intell Engineering College, Anantapur,AP, India

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

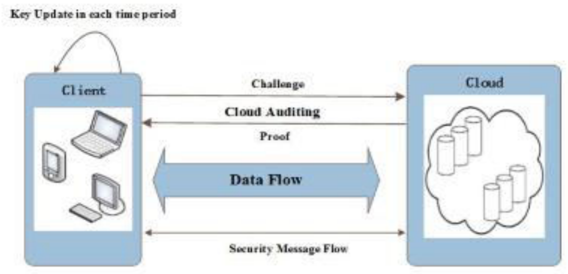
ENABLING CLOUD STORAGE WITH KEY EXPOSURE

[1]S.ARSHIYA SULTHANA, [2]P.NAMRATHA
[1]PG Scholar, CSE, Intell Engineering College, AP, India
[2]Associate Professor, CSE, Intell Engineering College, AP,India

ABSTRACT: Cloud storage auditing is viewed as an important service to verify the integrity of the data in public cloud. Current auditing protocols are all based on the assumption that the client's secret key for auditing is absolutely secure. However, such assumption may not always be held, due to the possibly weak sense of security and/or low security settings at the client. If such a secret key for auditing is exposed, most of the current auditing protocols would inevitably become unable to work. In this paper, we focus on this new aspect of cloud storage auditing. We investigate how to reduce the damage of the client's key exposure in cloud storage auditing, and give the first practical solution for this new problem setting. We formalize the definition and the security model of auditing protocol with key-exposure resilience and propose such a protocol. In our design, we employ the binary tree structure and the pre-order traversal technique to update the secret keys for the client. We also develop a novel authenticator construction to support the forward security and the property of block less verifiability. The security proof and the performance analysis show that our proposed protocol is secure and efficient.

1. INTRODUCTION

Cloud storage auditing is used to verify the integrity of the data stored in public cloud, which is one of the important security techniques in cloud storage. In recent years, auditing protocols for cloud storage have attracted much attention and have been researched intensively. These protocols focus on several different aspects of auditing, and how to achieve high bandwidth and computation efficiency is one of the essential concerns. For that purpose, the Homomorphism Linear Authenticator (HLA) technique that supports block less verification is explored to reduce the overheads of computation and communication in auditing protocols, which allows the auditor to verify the integrity of the data in cloud without retrieving the whole data. Many cloud storage auditing protocols like have been proposed based on this technique. The privacy protection of data is also an important aspect of cloud storage auditing. In order to reduce the computational burden of the client, a third-party auditor (TPA) is introduced to help the client to periodically check the integrity of the data in cloud. However, it is possible for the TPA to get the client's data after it executes the auditing protocol multiple times. Auditing protocols are designed to ensure the privacy of the client's data in cloud. Another aspect having been addressed in cloud storage auditing is how to support data dynamic operations. We initiate the first study on how to achieve the key exposure resilience in the storage auditing protocol and propose a new concept called auditing protocol with key-exposure resilience. In such a protocol, any dishonest behaviors, such as deleting or modifying some client's data stored in cloud in previous time periods, can all be detected, even if the cloud gets the client's current secret key for cloud storage auditing. This very important issue is not addressed before by previous auditing protocol designs. We further formalize the definition and the security model of auditing protocol with key-exposure resilience for secure cloud storage.We design and realize the first practical auditing protocol with built-in key-exposure resilience for cloud storage. In order to achieve our goal, we employ the binary tree structure, seen in a few previous works on different cryptographic designs, to update the secret keys of the client. Such a binary tree structure can be considered as a variant of the tree structure used in the HIBE scheme. In addition, the preorder traversal technique is used to associate each node of a binary tree with each time period. In our detailed protocol, the stack structure is used to realize the pre-order traversal of the binary tree. We also design a novel authenticator supporting the forward security and the property of block less verifiability.



1. System model of our cloud storage auditing

2. LITERATURE SURVEY

As the previous section reveals various methodologies for enabling cloud storage auditing, but still there is a huge gap to meet the perfection. So, as a step towards this, this paper tried to grab many concepts so that a new and efficient system can be proposed. The detailed studies are as follows. In [1], a thorough survey of various methods of cloud storage auditing is performed. Few existent methods have been analyzed and the challenged faced have been described in order to make an efficient protocol. When we store the data, the different version of the data is also stored uniformly. Thus, for the minimization of storage overhead, [2] "delta encoding" was adopted wherein the differences between the versions was noted. A specific type of delta encoding, skip delta encoding was adopted to optimize the added cost of storing and retrieving the data. K. Yang et al.[3] introduced a framework for auditing data storage in the cloud and also proposed an efficient privacy-preserving auditing protocol. Furthermore, it was extended to support dynamic operations like addition, deletion or modification of data. [4]explains the method of auditing the service dynamically to verify the integrity of a non trustable and outsourced storage on the basis of fragment structure, random sampling, and index-hash table, which also supported updates to the data outsourced and anomaly detection time to time. In [5], the authors have tried to improve the existing proof of storage models by using Merkle Hash Tree (MHT) construction for block tag authentication. In [6], the authors have presented two provably-secure PDP schemes which are more efficient than the aforementioned solutions, even when compared with schemes that achieve weaker guarantees. Furthermore, [7] extends the previous work on data possession proofs by the Multiple Replica Provable Data Possession (MR-PDP) for a single copy of a file in a client/server storage system. [8]introduces a mechanism of storage integrity auditing which permits the end users to compute the cost along with achieving fast data error localization, i.e it identifies if any server misbehaves. However, for an efficient auditing, a much more secure cloud storage system was proposed which supported privacy-preserving public auditing and the results were extended so that TPA could perform audits for multiple users at the same time and also execute it efficiently. Thus, in all the above works the cloud storage auditing is tried to make more efficient in various ways. As we all are already aware that the public key and the secret key play an important role in the encryption and the decryption of the data. If the secret key is exposed, it may lead to data forging and can get in hands of any unauthorized user. [9] narrates an idea of public key encryption which uses the concept of Binary Tree Encryption (BTE) wherein there is a master public key associated with the tree. Every node has a corresponding secret key and to encrypt a message destined for a particular node, one uses both public key and the name of the target node. The ciphertext which comes as a result can then be decrypted using the secret key of the target node. Now, atleast one secret key is used to sign the message in the current time-period and then obtain the secret key for the next time-period. As in the typical signature scheme, the public key is stable for all time-periods. A verification scheme checks both the validity of the signature and its time-period [10]. The signature scheme is forward secure because it might happen that signature can be forged for the previous time period even if it has the current secret key. As we discussed regarding the encryption of the keys, [11] introduced the concept of key-insulated security whose aim was to lessen the damage caused by secret key exposure. This was needed as usually cryptographic computations are performed on insecure devices. Thus, in this paper model has been proposed wherein the secret key stored on the insecure device are refreshed at various time periods along with a physically secure device which already possess the master key. In this way, the authors have construct a (t,N)- key-insulated encryption scheme based on any (standard) public key encryption scheme.

3. PROPOSED SYSTEM

We firstly show two basic solutions for the key-exposure problem of cloud storage auditing before we give our core protocol. 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.

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.

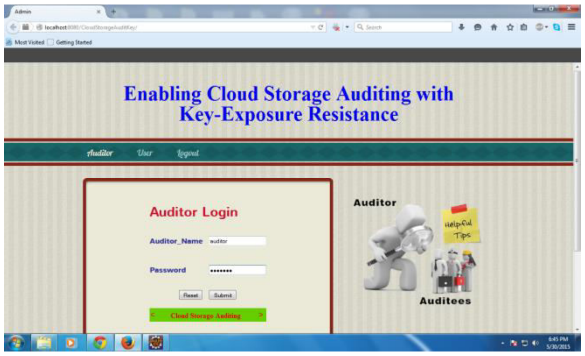
Slightly Better Solution

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 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 his storage. Then the new secret key is (SK j j+1 ,SK T ,· · · ,SK This solution is clearly better than the naive solution. Note j j from T).

Our Cloud Storage Auditing with Key-exposure Resilience:

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.

4.RESULTS



5.CONCLUSION

we study on how to deal with the client's key exposure in cloud storage auditing. We propose a new paradigm called auditing protocol with key-exposure resilience. In such a protocol, the integrity of the data previously stored in cloud can still be verified even if the client's current secret key for cloud storage auditing is exposed. We formalize the definition and the security model of auditing protocol with key-exposure resilience, and then propose the first practical solution. The security proof and the asymptotic performance evaluation show that the proposed protocol is secure and efficient.

REFERENCES

1. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security, pp. 598-609, 2007.

2. G. Ateniese, R.D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. 4th International Conference on Security and Privacy in Communication Networks, 2008

3. F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and J. Quisquater, "Efficient Remote Data In- tegrity checking in Critical Information Infrastructures," IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 8, pp. 1-6, 2008.

4. R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MRPPDP: Multiple-Replica Provable Data Possession," Proc. 28th IEEE International Conference on Distributed Computing Systems, pp. 411-420, 2008.

5. H. Shacham and B. Waters, "Compact Proofs of Retrievability," Advances in Cryptology-Asiacrypt'08, pp. 90-107, 2008.

6. C. Wang, K. Ren, W. Lou, and J. Li, "Toward Publicly Auditable Secure Cloud Data Storage Services," IEEE Network, vol. 24, no. 4, pp. 19-24, July/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," Proc. 17th ACM Conference on Computer and Communications Security, pp. 756-758, 2010.

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 and Distributed Systems, Vol. 24, No. 9, pp. 1717-1726, 2013.

10. C. Wang, S. Chow, Q. Wang, K. Ren, and W. Lou, “Privacy-Preserving Public Auditing for Secure Cloud Storage,” IEEE Trans. Computers, Vol. 62, No. 2, pp. 362375, 2013.