ENABLING CLOUD STORAGE AUDITING VIA VERIFIABLE KEY UPDATE OUTSOURCING

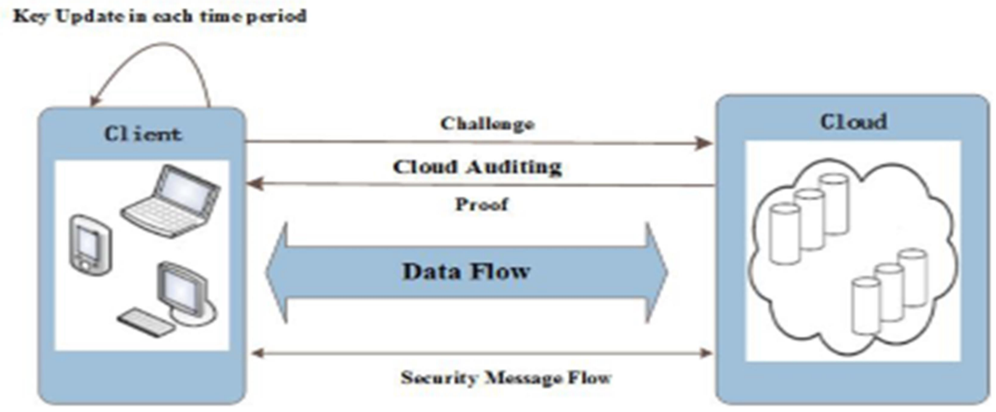
Dr. M SREENIVASULU*1, Y. KANAKA DURGA RAO*2, A JAGANMOHAN REDDY*3
***1 Professor, Dept. of Computer Science Engineering,**
****2, 3 Assistant Professor, Dept. of Computer Science Engineering.***
A.M Reddy Memorial College of Engineering and Technology, Andhra Pradesh

Abstract: Addressing the critical issue of key exposure resistance in cloud storage auditing has been a significant concern in security applications. Recent research has focused on finding solutions to this problem, which typically involve clients regularly updating their secret keys. However, this approach can be burdensome for clients with limited computational resources, such as mobile devices. To enhance client transparency during key updates, we propose a novel paradigm in this paper: cloud storage auditing combined with verifiable outsourcing of key updates. In our proposed model, clients can delegate the task of key updates to a trusted third party, alleviating the need for them to manage this process. We extend the role of the third-party auditor (TPA) from existing public auditing designs to include responsibility for both storage auditing and key security updates, thus preventing key exposure. Our scheme ensures that the TPA maintains only an encrypted copy of the client's secret key while performing these tasks on behalf of the client. During data transfer to the cloud, the client retrieves the encrypted secret key from the TPA. Additionally, our architecture provides mechanisms for the client to verify the authenticity of the encrypted secret keys supplied by the TPA. These features aim to make the auditing process with key exposure resistance as transparent as possible to the client. We formally outline the definition and underlying security model of this paradigm. Our implementations, rigorously tested and simulated, demonstrate the safety and effectiveness of the proposed designs in practice.
Keywords: Cloud storage, outsourcing computing, cloud storage auditing, key update, verifiability.

I. INTRODUCTION

One of the most crucial safety measures in cloud storage is auditing, which is implemented to ensure the authenticity of information kept in the cloud. Recent years have seen extensive studies into the topic of auditing protocols for cloud storage. One of the primary concerns of these protocols is how to achieve high bandwidth and computation efficiency, which is relevant to the auditing process as a whole. Therefore, the Homomorphism Linear Authenticator (HLA) technique is investigated, as it allows the auditor to verify the integrity of the data in the cloud without retrieving the whole data, thus reducing the computational and communication overheads of auditing protocols.
Many auditing protocols for cloud storage like have been proposed using this method. Auditing of cloud storage services should also take privacy measures seriously. A third-party auditor (TPA) is introduced to assist the client in performing periodic checks of the cloud's data integrity, thereby reducing the client's computational burden. After the TPA has executed the auditing protocol multiple times, it may be able to obtain the client's data. Client data in the cloud is kept private using auditing protocols. What's more, the auditing of cloud storage has addressed the question of how to facilitate data-dynamic operations.

In this project, we introduce the concept of an auditing protocol with key-exposure resilience and conduct the first research into its implementation for a storage auditing system. In such a protocol, even if the cloud acquires the client's current secret key for cloud storage auditing, any dishonest behavior, such as deleting or modifying some clients' data stored in the cloud in previous periods, can be detected. Existing designs for auditing protocols do not account for this crucial shortcoming. For safe cloud-based data storage, we formalize the definition and security model of auditing protocols with key-exposure resilience.
For the first time, we create a practical auditing protocol for cloud storage that includes built-in resistance to key exposure. The binary tree structure, used in a variety of earlier works on various cryptographic designs, is put to use here to accomplish our goal of keeping the client's secret keys secure and up to date. One could think of this binary tree structure as a subset of the tree structure employed by the HIBE scheme. Additionally, a binary tree's nodes are linked to their respective periods using the preorder traversal method. To implement the binary tree preorder traversal in our detailed protocol, we make use of the stack data structure.



1. System model of our cloud storage auditing

Using the system described below, we investigate the possibility of outsourcing numerical and logical computations. If a customer needs calculations performed but their internal resources (computer power, appropriate software, or programming skills) are insufficient, they may turn to an external operator for help. These days, we see this in a wide variety of everyday contexts, from fiscal to oil administrations. If the outsourcing is handled so that neither the original data nor the final calculation results are revealed to the third party, then it is considered a safe practice. The basic idea is that the client should perform some carefully planned local preprocessing (masking) of the problem or potentially information before sending it to the operator and that the client should also perform some carefully planned local postprocessing of the appropriate response come back to extract the true reply. As little effort as possible should be put into the camouflage process regarding the size of the information and the response. Numerical properties of computational performance may be altered by the camouflage preprocessing that the client performs locally to "conceal" the real calculation. Here, we present a structure for hiding logical computations and discuss their prices, numerical properties, and degrees of security. These covert operations can be set up in a normal state, a user-friendly framework (critical thinking condition), which conceals their complexity. We provide protocols for the secure and private outsourcing of direct polynomial math computations, allowing a client to safely outsource expensive logarithmic computations (such as the expansion of large-scale systems) to two remote servers while ensuring that neither server can learn anything about the client's private information or the result of the computation. The customer's local computations are efficient in the scope of their information and don't necessitate any time-consuming or money-sucking encryptions of the customer's input. The computational burden on the servers is proportional to the time unpredictability of the currently practically used calculations for addressing the arithmetic problem (e.g., relative to n3 for increasing two n x n networks). Even if the servers were to collude against the client, they could only learn where the client gets their information, but they couldn't change the correct response without the client knowing.

III. SYSTEM ANALYSIS

EXISTING SYSTEM

One of the most valued aspects of cloud computing is cloud storage. However, while cloud storage solves many problems for its users, it also introduces new security issues. How to efficiently verify the authenticity of cloud-based data is a pressing issue in information security. Many auditing protocols for cloud storage have been proposed to address this issue in recent years. Another critical issue with auditing cloud storage is the 'key exposure problem.'

Problems with the Current System:

1. Ineffectiveness of data integrity checks
2. If the cloud gains access to the client's secret key for storage auditing, it can easily cover up any data loss incidents, protect its reputation, and get rid of the client's infrequently used data to free up space.
3.

PROPOSED SYSTEM

A new paradigm is proposed, one that combines the auditing of cloud storage with the verifiable outsourcing of key updates. In the new model, the client no longer performs key-update operations but rather is an authorized third party. An authorized third party keeps the client's encrypted secret key and updates it each period for use in auditing cloud storage. If and when the client needs to upload new files to the cloud, he will need to download the encrypted secret key from the authorized party and decrypt it. The client also can validate the authenticity of the encrypted secret key. In this work, we develop the first auditing protocol for cloud storage that allows for the dependable delegation of key updates. Our architecture positions the TPA as the official custodian of all necessary revisions. Another crucial duty of the TPA is to verify the authenticity of the client's cloud-based data, much like the traditional public auditing protocols. Since the TPA only has an encrypted copy of the client's secret key, it cannot perform an audit of the client's cloud storage without access to the client's private key. To protect the TPA's private keys, we employ an encryption algorithm based on the blinding technique, which uses the homomorphic property. It improves the safety of our protocol and the speed of the decryption procedure. The TPA, meanwhile, can finish up any necessary key updates while in an encrypted state. After receiving the encrypted secret key from the TPA, the client can check its authenticity.

The Benefits of the Suggested Method:

1. The client is not aware of the TPA's handling of key updates in this protocol.
2. The TPA is only privy to the encrypted client secret key, and the client can double-check the TPA's encryption after it has downloaded the keys.

IV. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

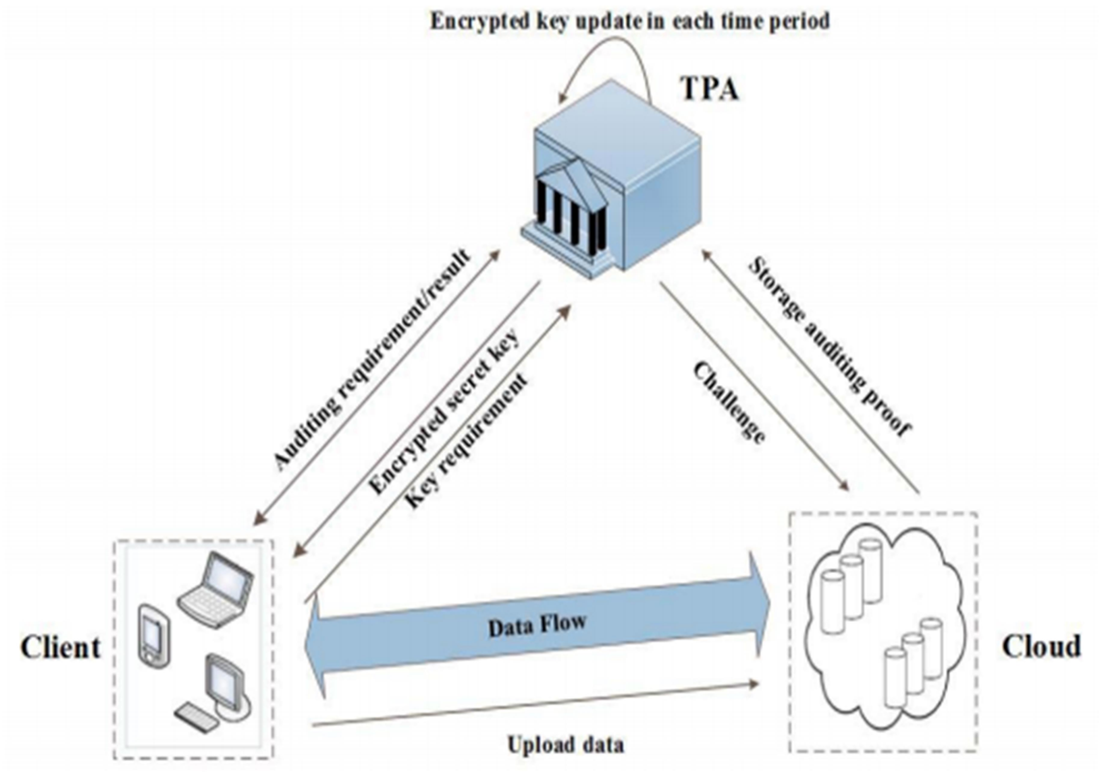


Fig. 1. System model of our cloud storage auditing.

MODULES

We have 3 main modules in this project;

1. Client Module
2. Cloud Module
3. Third Party Auditor (TPA)

Client:
The client retains all legal rights to any data stored in the cloud. Rather than having a set total size, the client can upload the expanding files to the cloud at any time.

Cloud:
The cloud serves as a repository for the client's data and a download hub.

TPA:
The TPA's primary function is to conduct an audit of the client's cloud-based data files, and its secondary function is to periodically update the client's encrypted secret keys.

V. CONCLUSION

To better protect our clients, we investigate the most vulnerable areas of their cloud storage accounts. An auditing protocol with key-exposure resilience is the new paradigm we propose. In such a protocol, even if the client's current secret key for cloud storage auditing is compromised, the data's integrity can still be verified for data that was previously stored in the cloud.We propose the first operational solution after formally defining and modelling the security of an auditing protocol that is resistant to key exposure. The proposed protocol is demonstrated to be safe and effective through proof of security and an analysis of its asymptotic performance.

VI. FUTURE WORK

The generation of the time period key is not something we suggest should be based on operations but on logging instead. It's inefficient to generate new keys all the time, so long periods of time between them are recommended instead. Automatically, based on a predetermined time period, a new key should be generated.

.

VII. REFERENCES

[1] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," Adv. Comput., vol. 54, pp. 215–272, 2002.
[2] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in Proc. 6th Annu. Conf. Privacy, Secur. Trust, 2008, pp. 240–245.
[3] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in Proc. IEEE INFOCOM, Apr. 2011, pp. 820–828.
[4] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in Proc. 17th Eur. Symp. Res. Comput. Secur., 2012, pp. 541–556.
[5] G. Ateniese et al., "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 598–609.
[6] A. Juels and B. S. Kaliski, Jr., "PORs: Proofs of retrievability for large files," in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 584–597.
[7] H. Shacham and B. Waters, "Compact proofs of retrievability," in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.
[8] 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.
[9] F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and 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.
[10] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiplereplica provable data possession," in Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst., Jun. 2008, pp. 411–420.