ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

A Framework to Enhance the Data Security in Cloud Storage Auditing With Key Abstraction

J.Navaneetha
Associate Professor
Department of CSE
Tirumala Engineering College.

H.K.Maheshwari
Associate Professor
Department of CSE
Tirumala Engineering College.

M.Paramesh
M.Tech Student
Department of CSE
Tirumala Engineering College.

ABSTRACT
To protect outsourced data in cloud storage against corruptions, adding fault tolerance to cloud storage together with data integrity checking and failure reparation becomes critical. Recently, regenerating codes have gained popularity due to their lower repair bandwidth while providing fault tolerance. Existing remote checking methods for regenerating-coded data only provide private auditing, requiring data owners to always stay online and handle auditing, as well as repairing, which is sometimes impractical. In this paper, we propose a public auditing scheme for the regenerating-code-based cloud storage. To solve the regeneration problem of failed authenticators in the absence of data owners, we introduce a proxy, which is privileged to regenerate the authenticators, into the traditional public auditing system model. Moreover, we design a novel public verifiable authenticator, which is generated by a couple of keys and can be regenerated using partial keys. Thus, our scheme can completely release data owners from online burden. In addition, we randomize the encode coefficients with a pseudorandom function to preserve data privacy. Extensive security analysis shows that our scheme is provable secure under random oracle model and experimental evaluation indicates that our scheme is highly efficient and can be feasibly integrated into the regenerating code- based cloud storage.

INTRODUCTION
Cloud storage is now gaining popularity because it offers a flexible on-demand data outsourcing service with appealing benefits: relief of the burden for storage management, universal data access with location independence, and avoidance of capital expenditure on hardware, software, and personal maintenances, etc.,. Nevertheless, this new paradigm of data hosting service also brings new security threats toward users data, thus making individuals or enterprisers still feel hesitant. It is noted that data owners lose ultimate control over the fate of their outsourced data; thus, the correctness, availability and integrity of the data are being put at risk. On the one hand, the cloud service is usually faced with a broad range of internal/external adversaries, who would maliciously delete or corrupt users' data; on the other hand, the cloud service providers may act dishonestly, attempting to hide data loss or corruption and claiming that the files are still correctly stored in the cloud for reputation or monetary reasons.

Thus it makes great sense for users to implement an efficient protocol to perform periodical verifications of their outsourced data to ensure that the cloud indeed maintains their data correctly. Many mechanisms dealing with the integrity of outsourced data without a local copy have been proposed under different system and security models up to now. The most significant work among these studies are the PDP (provable data possession) model and POR (proof of retrievability) model, which were originally proposed for the single-server scenario by Ateniese et al. and Juels and Kaliski, respectively. Considering that files are usually striped and redundantly stored across multi-servers or multi-clouds, explore integrity verification schemes suitable for such multi-servers or multi-clouds setting with different redundancy schemes, such as replication, erasure codes, and, more recently, regenerating codes

In this paper, we focus on the integrity verification problem in regenerating-code-based cloud storage, especially with the functional repair strategy. Similar studies have been performed by Chen et al. and Chen and Lee separately and independently. extended the single-server CPOR scheme to the regeneratingcode-scenario; designed and implemented a data integrity protection scheme for FMSR -based cloud storage and the scheme is adapted to the thin-cloud setting.1 However, both of them are designed for private audit, only the data owner is allowed to verify the integrity and repair the faulty servers. Considering the large size of the outsourced data and the user's constrained resource capability, the tasks of auditing and reparation in the cloud can be formidable and expensive for the users.

The overhead of using cloud storage should be minimized as much as possible such that a user does not need to perform too many operations to their outsourced data (in additional to retrieving it). In particular, users may not want to go through the complexity in verifying and reparation. The auditing schemes in and imply the problem that users need to always stay online, which may impede its adoption in practice, especially for long-term archival storage. To fully ensure the data integrity and save the users' computation resources as well as online burden, we propose a public auditing scheme for the regenerating-code-based cloud storage, in which the integrity checking and regeneration (of failed data blocks and authenticators) are implemented by a third-party auditor and a semi-trusted proxy separately on behalf of the data owner. Instead of directly adapting the existing public auditing scheme to the multi-server setting, we design a novel authenticator, which is more appropriate for regenerating codes.

EXISTING SYSTEM:

Cloud storage is now gaining popularity because it offers a flexible on-demand data outsourcing service with appealing benefits: relief of the burden for storage management, universal data access with location independence, and avoidance of capital expenditure on hardware, software, and personal maintenances.

DISADVANTAGES:

It is noted that data owners lose ultimate control over the fate of their outsourced data; thus, the correctness, availability and integrity of the data are being put at risk.

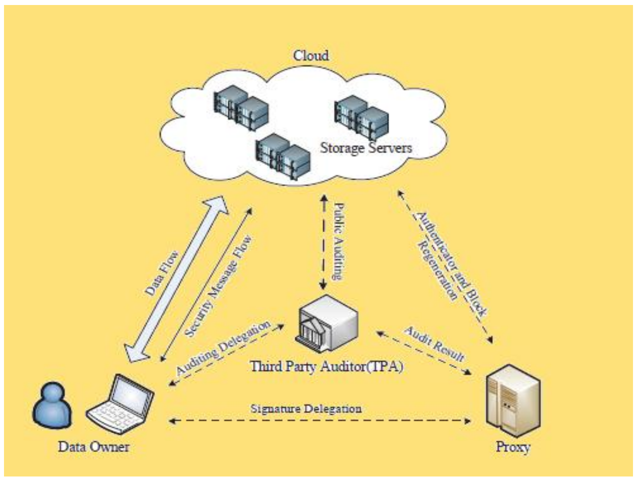
PROPOSED SYSTEM:

The integrity of outsourced data without a local copy have been proposed under different system and security models up to now. The most significant work among these studies are the PDP (provable data possession) model and POR (proof of retrievability) model, which were originally proposed for the single-server scenario by Considering that files are usually striped and redundantly stored across multi-servers or multi-clouds, explore integrity verification schemes suitable for such multi-servers or multi clouds setting with different redundancy schemes, such as replication, erasure codes, and, more recently, regenerating codes.

ADVANTAGES:

We focus on the integrity verification problem in regenerating-code-based cloud storage, especially with the functional repair strategy.

SYSTEM ARCHITECTURE:



IMPLEMENTATION MODULES:

1. Regenerating Codes:
2. Design Goals
3. Definitions of Our Auditing Scheme
4. Enabling Privacy-Preserving Auditable

MODULES DESCRIPTION:

Regenerating Codes:

Regenerating codes are first introduced for distributed storage to reduce the repair bandwidth. Viewing cloud storage to be a collection of n storage servers, data file F is encoded and stored redundantly across these servers. Then F can be retrieved by connecting to any k-out-of-n servers, which is termed the MDS2-property. When data corruption at a server is detected, the client will contact ℓ healthy servers and download β′ bits from each server, thus regenerating the corrupted blocks without recovering the entire original file.

Design Goals:

To correctly and efficiently verify the integrity of data and keep the stored file available for cloud storage, our proposed auditing scheme should achieve the following properties:

- Public Auditability: to allow TPA to verify the intactness of the data in the cloud on demand without introducing additional online burden to the data owner.
- Storage Soundness: to ensure that the cloud server can never pass the auditing procedure except when it indeed manage the owner's data intact.
- Privacy Preserving: to ensure that neither the auditor nor the proxy can derive users' data content from the auditing and reparation process.
- Authenticator Regeneration: the authenticator of the repaired blocks can be correctly regenerated in the absence of the data owner.
- Error Location: to ensure that the wrong server can be quickly indicated when data corruption is detected.

Definitions of Our Auditing Scheme

Our auditing scheme consists of three procedures: Setup, Audit and Repair. Each procedure contains certain polynomial-time algorithms as follows:

Setup: The data owner maintains this procedure to initialize the auditing scheme. KeyGen(1κ) → (pk, sk): This polynomial-time algorithm is run by the data owner to initialize its public and secret parameters by taking a security parameter κ as input.

Degelation(sk) → (x): This algorithm represents the interaction between the data owner and proxy. The data owner delivers partial secret key x to the proxy through a secure approach.

Sig And BlockGen (sk, F) → (_,, t): This polynomial time algorithm is run by the data owner and takes the secret parameter sk and the original file F as input, and then outputs a coded block set , an authenticator set _ and a file tag t.

Audit: The cloud servers and TPA interact with one another to take a random sample on the blocks and check the data intactness in this procedure.

Challenge(Finfo) → (C): This algorithm is performed by the TPA with the information of the file Finfo as input and a challenge C as output.

ProofGen(C,_,) → (P): This algorithm is run by each cloud server with input challenge C, coded block set and authenticator set _, then it outputs a proof P.

Verify(P, pk, C) → (0, 1): This algorithm is run by TPA immediately after a proof is received. Taking the proof P, public parameter pk and the corresponding challenge C as input, it outputs 1 if the verification passed and 0 otherwise.

Repair: In the absence of the data owner, the proxy interacts with the cloud servers during this procedure to repair the wrong server detected by the auditing process.

Enabling Privacy-Preserving Auditable:

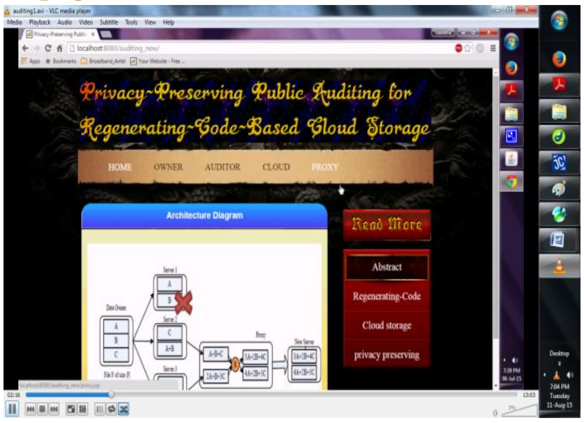
The privacy protection of the owner's data can be easily achieved through integrating with the random

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

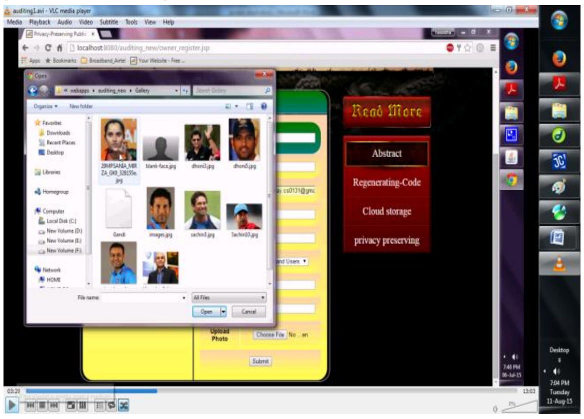
proof blind technique or other technique . However, all these privacy-preservation methods introduce additional computation overhead to the auditor, who usually needs to audit for many clouds and a large number of data owners; thus, this could possibly make it create a performance bottleneck. Therefore, we prefer to present a novel method, which is more light-weight, to mitigate private data leakage to the auditor. Notice that in a regenerating-code-based cloud storage, data blocks stored at servers are coded as linear combinations of the original blocks Supposing that the curious TPA has recovered m coded blocks by elaborately performing Challenge-Response procedures and solving systems of linear equations], the TPA still requies to solve another group of m linearly independent equations to derive the m native blocks.

SCREEN SHOTS

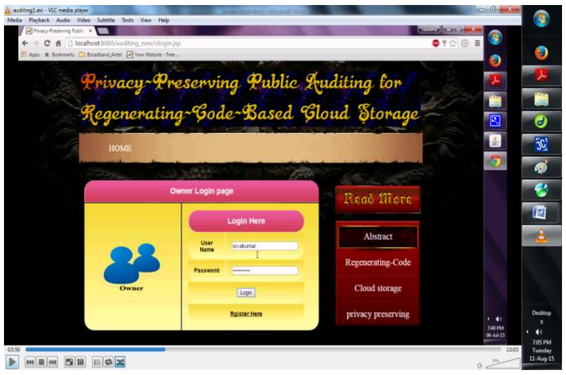
Homepage:


Privacy-Preserving Public Auditing for Regenerating-Code-Based Cloud Storage
HOME OWNER AUDITOR CLOUD PROXY
Architecture Diagram
Read More
Abstract
Regenerating-Code
Cloud storage
privacy preserving

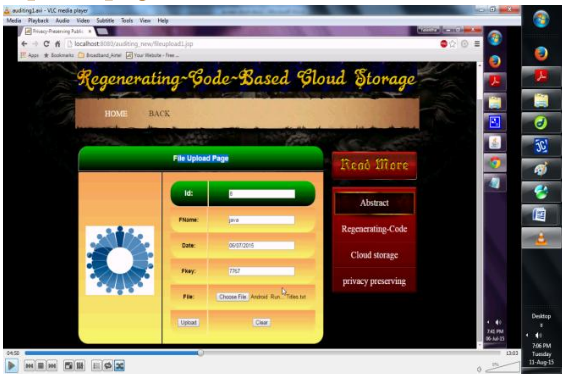
Registration page:



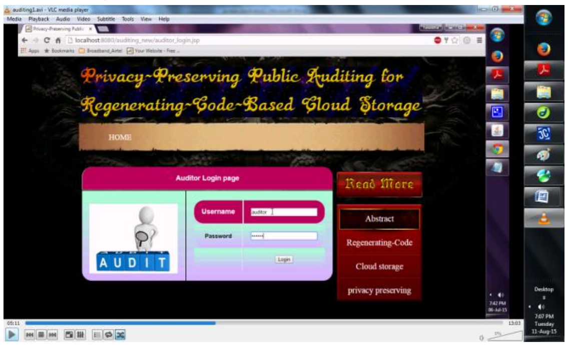
Owner login:



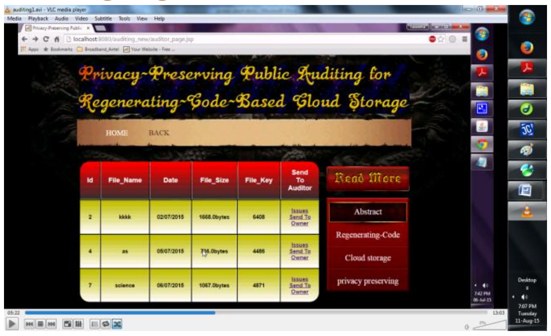
File upload page:



Auditor Login:



File Auditing Page:


Privacy-Preserving Public Auditing for Regenerating-Code-Based Cloud Storage
HOME BACK
Read More
Abstract
Regenerating-Code
Cloud storage
privacy preserving

Id	File_Name	Date	File_Size	File_Key	Send To Auditor
2	kkkk	02/07/2015	1666.0bytes	6408	Issues Send To Owner
4	aa	06/07/2015	716.0bytes	4486	Issues Send To Owner
7	science	06/07/2015	1067.0bytes	4871	Issues Send To Owner

CONCLUSION

We propose a public auditing scheme for the regenerating-code-based cloud storage system, where the data owners are privileged to delegate TPA for their data validity checking. To protect the original data privacy against the TPA, we randomize the coefficients in the beginning rather than applying the blind technique during the auditing process. Considering that the data owner cannot always stay online in practise, in order to keep the storage available and verifiable after a malicious corruption, we introduce a semi-trusted proxy into the system model and provide a privilege for the proxy to handle the reparation of the coded blocks and authenticators. To better appropriate for the regenerating-code-scenario, we design our authenticator based on the BLS signature. This authenticator can be efficiently generated by the data owner simultaneously with the encoding procedure. Extensive analysis shows that our scheme is provable secure, and the performance evaluation shows that our scheme is highly efficient and can be feasibly integrated into a regenerating-code-based cloud storage system.

REFERENCES

[1] M. Armbrust et al., “Above the clouds: A Berkeley view of cloud computing,” Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2009-28, 2009.

[2] G. Ateniese et al., “Provable data possession at untrusted stores,” in Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS), New York, NY, USA, 2007, pp. 598–609.

[3] 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.

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

[5] K. D. Bowers, A. Juels, and A. Oprea, “HAIL: A high-availability and integrity layer for cloud storage,” in Proc. 16th ACM Conf. Comput. Commun. Secur., 2009, pp. 187–198.

[6] J. He, Y. Zhang, G. Huang, Y. Shi, and J. Cao, “Distributed data possession checking for securing multiple replicas in geographicallydispersed clouds,” J. Comput. Syst. Sci., vol. 78, no. 5, pp. 1345–1358, 2012.

[7] B. Chen, R. Curtmola, G. Ateniese, and R. Burns, “Remote data checking for network coding-based distributed storage systems,” in Proc. ACM Workshop Cloud Comput. Secur. Workshop, 2010, pp. 31–42.

[8] H. C. H. Chen and P. P. C. Lee, “Enabling data integrity protection in regenerating-coding-based cloud storage: Theory and implementation,” IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 407–416, Feb. 2014.

[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] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, “Cooperative provable data possession for integrity verification in multicloud storage,” IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 12, pp. 2231–2244, Dec. 2012.

[11] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, “A survey on network codes for distributed storage,” Proc. IEEE, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[12] H. Shacham and B. Waters, “Compact proofs of retrievability,” in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.

[13] Y. Hu, H. C. H. Chen, P. P. C. Lee, and Y. Tang, “NCCloud: Applying network coding for the storage

repair in a cloud-of-clouds," in Proc. USENIX FAST, 2012, p. 21.

[14] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in Proc. IEEE INFOCOM, Mar. 2010, pp. 1–9.

[15] 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.