International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-3, Issue-11)

Research Article | **November 2014**

An Idea on Enhanced Security Based Approach for Restoring Data and Preserving its Integrity by using Regenerating Erasure Codes

V. Anto Vins, Mary Leema Rose. A, Uma Mageswari .S
PITAM,
Tamil Nadu India

Abstract—

Cloud storage mainly offers an on-demand data outsourcing servicing model which is gaining popularity through its elasticity and low maintenance cost. In such cases Security concerns arises when data is outsourced to a third-party cloud storage provider. The data that is stored in the cloud might get corrupted due to some malicious programs or due to an adversary. Considering the existing methodology FMSR-DIP codes which take the random subsets of data to check data integrity and by using TTPA regenerate the corrupted data. While regenerating the corrupted data it is only possible to restore the entire data block which is time-consuming and provides less scalability. To resolve such an issue the proposed system depicts the method in which the data uploaded get's double-encrypted and stored onto the main server. The encrypted data are separated into chunks and placed in replica server's and then converted into binary format. After adding parity bits to the binary format they are stored in the sub-servers and are hashed to produce a hash signature. They are monitored periodically by TTPA to check integrity and if data loss or corruptions are detected, they are regenerated by using erasure codes with less repair traffic and low time-consumption. Hence this paper revolves around the main objective of using an erasure coding technique that minimizes the time consumption and preserving data confidentiality and integrity. It confirms to provide a way to ensure that the trusted third-party auditor cannot corrupt the data block during periodical monitoring.

Keywords— Erasure codes, Regenerating codes, Repair traffic, Fault tolerance, TTPA.

I. INTRODUCTION

To protect outsourced data in cloud storage against corruptions, adding fault tolerance to cloud storage, along with efficient data integrity checking and recovery procedures, becomes critical. Regenerating codes provide fault tolerance by striping data across multiple servers, while using less repair traffic than traditional erasure codes during failure recovery. Therefore, the study of problem of remotely checking the integrity of regenerating-coded data against corruptions under a real-life cloud storage setting uses the design and implement of a practical data integrity protection (DIP) scheme for a specific regenerating code, while preserving its intrinsic properties of fault tolerance and repair-traffic saving. It works under the simple assumption of thin-cloud storage and allows different parameters to be fine-tuned for a performance-security trade-off. One major use of cloud storage is long-term archival, which represents a workload that is written once and rarely read. While the stored data are rarely read, it remains necessary to ensure its integrity for disaster recovery or compliance with legal requirements. Since it is typical to have a huge amount of archived data, whole-file checking becomes prohibitive. Proof of retrievability (POR) and proof of data possession (PDP) have thus been proposed to verify the integrity of a large file by spot checking only a fraction of the file via various cryptographic primitives.

II. EXISTING SYSTEM

In the existing system un-encoded data has been been used by FMSR codes which provides less security and allows the third party auditor to become a malicious attacker. In the proposed system double-encryption is done and stored to the replica server. The encrypted data is binary converted and a parity bit is added , at the same time XOR of the block occurs and the XORed value of each block is stored in the replica as well as a copy of it is given to the main server. In the proposed system hash values are generated for the blocks and given to TTPA which disables the direct access of data by the TTPA. Three main operations involved in the existing system includes the following

a. Upload Operation

Step 1: Generate the per-file secrets.
Step 2: Encode the file using FMSR codes.
Step 3: Encode each code chunk with FMSR-DIP codes.
Step 4: Update the metadata file and upload.

b. Check Operation

In the Check operation, we verify randomly chosen rows of bytes of the currently stored chunks in the servers.
Step 1: Check the metadata file. We download a copy of the encrypted metadata from each server and check if all copies are identical.

Step 2: Sampling and row verification. Based on the FMSRDIP code chunk size b' , we randomly generate $\lfloor \lambda b' \rfloor$distinct indices, where $\lambda \epsilon (0,1]$ is a tunable checking percentage.
Step 3: Error localization Step 4: Trigger repair. If a server has more than a user specified number of bytes marked as corrupted, we consider it a failed server and trigger the Repair operation.

c. Download Operation

We download a file F from the servers as follows: Step 1: Check the metadata file. Refer to Step 1 of Check.
Step 2: Download and decode the FMSR-DIP code chunks for file F. To reconstruct file F, we download $k(n-k)$FMSR-DIP code chunks from any k servers (without the AECC parities). After downloading a code chunk, we verify its integrity with the corresponding MAC. We strip the PRFs off the FMSR-DIP code chunks to form the FMSR code chunks, which are then passed to NC Cloud for decoding if they are not corrupted. However, if we have a corrupted code chunk, then we can fix it with one of the following approaches:

- Download its AECC parities and apply error correction. Then we verify the corrected chunk with its MAC again.
- Download the$(n-k)$code chunks from another server.
- A last resort is to download the code chunks from all n servers. We check all rows of the chunks including their AECC parities. The rows with a subset of the bytes marked correct can be recovered with FMSR codes; the rows with all bytes marked corrupted are treated as erasures and will be corrected with AECC.A file is deemed unrecoverable if there are insufficient code chunks that pass their MAC verifications.

d. Repair Operation

If some server fails (e.g., when losing all data or having too much corrupted data that cannot be recovered), we trigger the Repair operation via NC Cloud as follows:
Step 1: Check the metadata file. Step 3: Encode, update metadata, and upload.
Step 2: Download and decode the needed chunks.

III. PROPOSED SYSTEM

In the proposed system, the user register's himself with the cloud storage service provider. To enhance security, when the data is uploaded double-encryption(i.e by using R.S.A and A.E.S) takes place. Now the data is spitted and transferred across the replica servers. In the replica servers the encrypted data is converted to binary format and a parity bit is added to it for error detection. This binary data is sent to the multi-server setting where the erasure coding mechanism XOR's the data in one server with the other to get the erasure codes for regenerating the original data if the data get's corrupted. In the existing system the trusted third-party auditor is given a copy of the codes to check the data integrity, there arises a situation where the third-party auditor might become a threat to the process. In the proposed system, the data in each server of the multi-server setting is hashed by using SHA-512 to generate a unique hash-code for each block of data. This hash-code generated for each server is given to the third-party auditor. At regular intervals the data in each server of the multi-server setting is hashed and the third-party auditor checks the integrity by comparing the hash codes. If any deviation in the hash-code is monitored, then the data might be corrupted or changed by a malicious hacker. The trusted third-party auditor informs that the data is corrupted to the main server and the main server retrieve's the data from the replica server's, if the data in the replica is also corrupted then erasure coding mechanism is invoked on the servers to regenerate the corrupted data.

IV. SYSTEM ARCHITECTURE

The Outline of the System Architecture revolves on reducing the Repair traffic with the process of checking whether the data is corrupted and regenerating the corrupted part of the data with the help of these modules. First module is the Main Server that registers the Cloud user and allows the user to upload their data. Third party auditor checks the hash code of each server in the Multi Server setting, if there is an deviation in the hash codes the auditor indicates to the main cloud server. Main Cloud Server checks the data originality in the Replica Server and invokes Erasure Coding mechanism if it has defect in Replica Server too.

Main module is the Replica server that transforms the encrypted data into binary bits and adds parity bits to it. The Multi-server setting hashes the data from the Replica server output and invokes the Erasure coding mechanism.

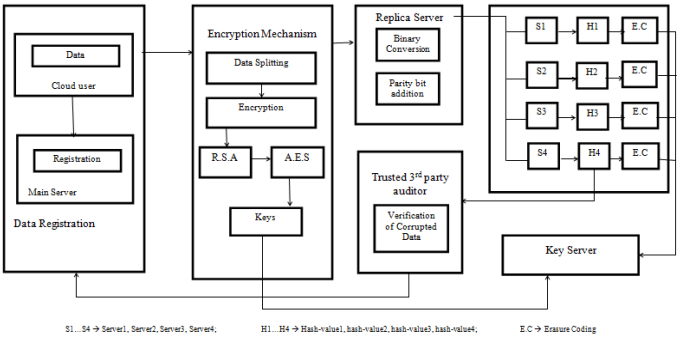

Fig. 1: Repair Traffic & Data Integrity Architecture

V. CONCLUSION

The Effectiveness of the repair traffic is maintained and the bandwidth consumption is limited. The data can be restored without much time consumption. This binary data is sent to the multi-server setting where the Erasure coding mechanism XOR's the Data in one server with the other to get the erasure codes for regenerating the original data if the data get's corrupted. In the existing system the trusted third-party auditor is given a copy of the codes to check the data integrity, there arises a situation where the third-party auditor might become a threat to the process. In the proposed system, the data in each server of the multi-server setting is hashed by using SHA-512 to generate a unique hash-code for each block of data. This hash-code generated for each server is given to the third-party auditor. At regular intervals the data in each server of the multi-server setting is hashed and the third-party auditor checks the integrity by comparing the hash codes. If any deviation in the hash-code is monitored, the data might be corrupted or changed by a malicious hacker. This provides an effective way for enhanced data integrity protection in the cloud servers and restore the data back at minimal cost.

REFERENCES

[1] Henry C.H. Chen and Patrick P.C. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage: Theory and Implementation", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, Vol. 25, No. 2, February 2014.

[2] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon, "RACS: A Case for Cloud Storage Diversity,"Proc. First ACM Symp. Cloud Computing (SoCC '10),2010.

[3] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing,"Comm. ACM,vol. 53, no. 4, pp 50-58, 2010.

[4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, "Remote Data Checking Using Provable Data Possession,"ACM Trans. Information and System Security,vol. 14, article 12, May 2011.

[5] K. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09),2009.

[6] K. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation,"Proc. ACM Workshop Cloud Computing Security (CCSW '09), 2009.

[7] B. Chen, R. Curtmola, G. Ateniese, and R. Burns, "Remote Data Checking for Network Coding-Based Distributed Storage Systems,"Proc. ACM Workshop Cloud Computing Security (CCSW '10), 2010.

[8] H.C.H. Chen and P.P.C. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage," Proc. IEEE 31stSymp. Reliable Distributed Systems (SRDS '12),2012.

[9] L. Chen, "NIST Special Publication 800-108,"Recommendation for Key Derivation Using Pseudorandom Functions (Revised), http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf, Oct.2009.

[10] R. Curtmola, O. Khan, and R. Burns, "Robust Remote Data Checking,"Proc. ACM Fourth Int'l Workshop Storage Security and Survivability (StorageSS '08),2008.

[11] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-Replica Provable Data Possession,"Proc. IEEE 28th Int'l Conf. Distributed Computing Systems (ICDCS '08),2008.

[12] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," IEEE Trans. Information Theory, vol. 56, no. 9, 4539-4551, Sept. 2010.

[13] D. Ford, F. Labelle, F.I. Popovici, M. Stokel, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in Globally Distributed Storage Systems,"Proc. Ninth USENIX Symp. Operating Systems Design and Implementation (OSDI '10),Oct. 2010.

[14] O. Goldreich,Foundations of Cryptography: Basic Tools.CambridgeUniv. Press, 2001.

[15] O. Goldreich,Foundations of Cryptography: Basic Applications. Cambridge Univ. Press, 2004.

[16] Y. Hu, H. Chen, P. Lee, and Y. Tang, "NCCloud: Applying Network Coding for the Storage Repair in a Cloud-of-Clouds," Proc. 10th USENIX Conf. File and Storage Technologies (FAST '12),2012.

[17] A. Juels and B. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files,"Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07),2007.

[18] H. Krawczyk, "Cryptographic Extraction and Key Derivation: The HKDF Scheme,"Proc. 30th Ann. Conf. Advances in Cryptology (CRYPTO '10),2010.

[19] E. Naone, "Are We Safeguarding Social Data?" http://www.technologyreview.com/blog/editors/22924/, Feb. 2009.

[20] J.S. Plank, "A Tutorial on Reed-Solomon Coding for FaultTolerance in RAID-Like Systems,"Software - Practice & Experience,vol. 27, no. 9, pp. 995-1012, Sept. 1997.

[21] M.O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance,"J. ACM, vol. 36, no. 2, pp. 335-348, Apr. 1989.

[22] I. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields,"J. Soc. Industrial and Applied Math., vol. 8, no. 2, pp. 300-304, 1960.

[23] B. Schroeder, S. Damouras, and P. Gill, "Understanding Latent Sector Errors and How to Protect against Them,"Proc. USENIX Conf. File and Storage Technologies (FAST '10),Feb. 2010.

[24] B. Schroeder and G.A. Gibson, "Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You?"Proc. Fifth USENIX Conf. File and Storage Technologies (FAST '07),Feb. 2007.

[25] T. Schwarz and E. Miller, "Store, Forget, and Check: UsingAlgebraic Signatures to Check Remotely Administered Storage," Proc. IEEE 26th Int'l Conf. Distributed Computing Systems, (ICDCS '06), 2006.

[26] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '08),2008.

[27] "TechCrunch,"Online Backup Company Carbonite Loses Customers' Data, Blames and Sues Suppliers,http://techcrunch.com/2009/03/23/online-backup-company-carbonite-loses-customers-datablames-and-sues-suppliers/, Mar. 2009.

[28] M. Vrable, S. Savage, and G. Voelker, "Cumulus: FilesystemBackup to the Cloud," Proc. USENIX Conf. File and Storage Technologies (FAST),2009.

[29] "Watson Hall Ltd," UK Data Retention Requirements,https://www.watsonhall.com/resources/downloads/paper-uk-dataretention-requirements.pdf, 2009.

[30] A. Wildani, T.J.E. Schwarz, E.L. Miller, and D.D. Long, "Protecting Against Rare Event Failures in Archival Systems,"Proc. IEEE Int'l Symp. Modeling, Analysis and Simulation Computer and Telecomm. Systems (MASCOTS '09),2009.

