© 2016 IJSRSET | Volume 2 | Issue 2 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Public Auditing for Regeneration Code Based Cloud Storage Using Homomorphic Encryption for User Privacy

Sai Krishnan R, Rajasekar E, Divya S

Computer Science and Engineering, Dhanalakshmi College of Engineering, Tambaram, Chennai, Tamil Nadu, India

ABSTRACT

To protect the outsourced data in cloud storage against corruptions, adding fault tolerance to cloud storage together with data integrity checking and failure reparation becomes critical. Existing remote checking methods for regenerating-coded data only provide public auditing with the help of Third Party Auditor (TPA) and Proxy to manage and recover the data if lost, but there is a lack of user privacy. This is solved by using homomorphic encryption. Homomorphic encryption is the conversion of data into cipher text that can be analysed and worked with as if it were still in its original form. It allows complex mathematical operations to be performed on encrypted data without compromising the encryption thus providing an additional layer of user level security.

Keywords: Homomorphic Encryption, Public Auditing, Regeneration Code

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers.

The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power large, immersive computer games.

The cloud computing uses networks of large groups of servers typically running low-cost consumer PC technology with specialized connections to spread data processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing.

Characteristics and Services Models:

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

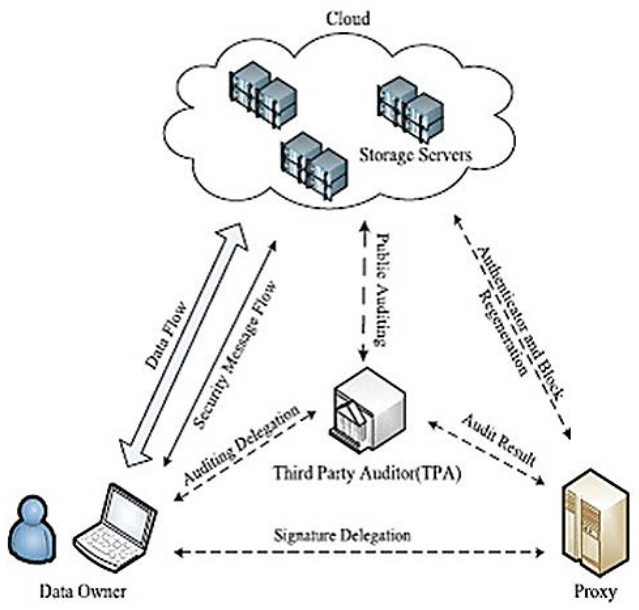
- **On-demand self-service**: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.
- **Broad network access**: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).
- **Resource pooling**: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the

customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.

- **Rapid elasticity**: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

II. METHODS AND MATERIAL

A. System Architecture



B. System Model

We consider the auditing system model for Regenerating-Code-based cloud storage, which involves four entities: *the data owner*, who owns large amounts of data files to be stored in the cloud; *the cloud*, which are managed by the cloud service provider, provide storage service and have significant computational resources; *the third party auditor* (TPA), who has expertise and capabilities to conduct public audits on the coded data in the cloud, the TPA is trusted and its audit result is unbiased for both data owners and cloud servers; and *a proxy agent*, who is semi-trusted and acts on behalf of the data owner to regenerate authenticators and data blocks on the failed servers during the repair procedure. Notice that the data owner is restricted in computational and storage resources compared to other entities and may become off-line even after the data upload procedure. The proxy, who would always be online, is supposed to be much more powerful than the data owner but less than the cloud servers in terms of computation and memory capacity. To save resources as well as the online burden potentially brought by the periodic auditing and accidental repairing, the data owners resort to the TPA for integrity verification and delegate the reparation to the proxy.

C. Construction of Our Auditing Scheme

Our auditing scheme consists of three procedures: Setup, Audit and Repair. 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 manages 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.

D. Mitigating the Overhead of Data Owner

Despite that the data owner has been released from online burden for auditing and repairing, it still makes sense to reduce its computation overhead in the Setup phase because data owners usually maintain very limited computational and memory resources. As previously described, authenticators are generated in a new method which can reduce the computational complexity of the owner to some extent; however, there exists a much more efficient method to introduce further reduction. Considering that there are so many modular exponent arithmetic operations during the authenticator generation, the data owner can securely delegate part of its computing task to the proxy in the following way: The data owner first properly augments the *m* native blocks, signs for them, and thus obtains and, then it sends the augmented native blocks and to the proxy. After receiving from the data owner, the proxy implements the last two steps of *SigAndBlockGen(·)* and finally generates entire authenticators for each segment with secret value *x*. In this way, the data owner can migrate the expensive encoding and authenticator generation task to the proxy while itself maintaining only the first two lightweight steps; thus, the workload of data owner can be greatly mitigated.

III. RESULTS AND DISCUSSION

Enabling Privacy-Preserving Auditable

The privacy protection of the owner’s data can be easily achieved through integrating with the random 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 regenerating-code-based cloud storage, data blocks stored at servers are coded as linear combinations of the original blocks with random coefficients.

1) Above the Clouds: A Berkeley View of Cloud Computing

Provided certain obstacles are overcome, we believe Cloud Computing has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new interactive Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about over-provisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or under-provisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get their results as quickly as their programs can scale, since using 1000 servers for one hour costs no more than using one server for 1000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT. The economies of scale of very large-scale data centres combined with ``pay-as-you-go" resource usage has heralded the rise of Cloud Computing. It is now attractive to deploy an innovative new Internet service on a third party's Internet Datacenter rather than your own infrastructure, and to gracefully scale its resources as it grows or declines in popularity and revenue. Expanding and shrinking daily in response to normal diurnal patterns could lower costs even further. Cloud Computing transfers the risks of over-provisioning or under-provisioning to the Cloud Computing provider, who mitigates that risk by statistical multiplexing over a much larger set of users and who offers relatively low prices due better utilization and from the economy of purchasing at a larger scale. We define terms, present an economic model that quantifies the key buy vs. pay-as-you-go decision, offer a spectrum to classify Cloud Computing providers, and give our view of the top 10 obstacles and opportunities to the growth of Cloud Computing.

2) Provable Data Possession at Untrusted Stores

We introduce 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. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage system.

We present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

3) PORs: Proofs of Retrievability for Large Files

In this paper, we define and explore *proofs of retrievability* (PORs). A POR scheme enables an archive or back-up service (prover) to produce a concise proof that a user (verifier) can retrieve a target file *F*, that is, that the archive retains and reliably transmits file data sufficient for the user to recover *F* in its entirety.

A POR may be viewed as a kind of cryptographic proof of knowledge (POK), but one specially designed to handle a *large* file (or bit string) *F*. We explore POR protocols here in which the communication costs, number of memory accesses for the prover, and storage requirements of the user (verifier) are small parameters essentially independent of the length of *F*.

In addition to proposing new, practical POR constructions, we explore implementation considerations and optimizations that bear on previously explored, related schemes. In a POR, unlike a POK, neither the prover nor the verifier need actually have knowledge of *F*. PORs give rise to a new and unusual security definition whose formulation is another contribution of our work.

We view PORs as an important tool for semi-trusted online archives. Existing cryptographic techniques help users ensure the privacy and integrity of files they retrieve. It is also natural, however, for users to want to verify that archives do not delete or modify files prior to retrieval. The goal of a POR is to accomplish these checks *without users having to download the files themselves*. A POR can also provide quality-of-service guarantees, i.e., show that a file is retrievable within a certain time bound. The cloud formatted to be easy.

4) MR-PDP: Multiple-Replica Provable Data Possession

Many storage systems rely on replication to increase the availability and durability of data on untrusted storage systems. At present, such storage systems provide no strong evidence that multiple copies of the data are actually stored. Storage servers can collude to make it look like they are storing many copies of the data, whereas in reality they only store a single copy. We address this shortcoming through multiple-replica provable data possession (MR-PDP): A provably-secure scheme that allows a client that stores t replicas of a file in a storage system to verify through a challenge-response protocol that (1) each unique replica can be produced at the time of the challenge and that (2) the storage system uses t times the storage required to store a single replica. MR-PDP extends previous work on data possession proofs for a single copy of a file in a client/server storage system (Ateniese et al., 2007). Using MR-PDP to store t replicas is computationally much more efficient than using a single-replica PDP scheme to store t separate, unrelated files (e.g., by encrypting each file separately prior to storing it). Another advantage of MR-PDP is that it can generate further replicas on demand, at little expense, when some of the existing replicas fail.

IV. CONCLUSION

In this paper, 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.

V. 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 geographically- dispersed 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-basedcloudstorage:Theoryandimplementation,” 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.

[16] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, “Toward secure and dependable storage services in cloud computing,” IEEE Trans. Service Comput., vol. 5, no. 2, pp. 220–232, Apr./Jun. 2012.

[17] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the Weil pairing,” J. Cryptol., vol. 17, no. 4, pp. 297–319, 2004.

[18] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” IEEE Trans. Inf. Theory, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[19] T. Ho et al., “A random linear network coding approach to multicast,” IEEE Trans. Inf. Theory, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[20] D. Boneh, D. Freeman, J. Katz, and B. Waters, “Signing a linear subspace: Signature schemes for network coding,” in Public Key Cryptography. Berlin, Germany: Springer-Verlag, 2009, pp. 68–87.

[21] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 2001, pp. 213–229.

[22] A. Miyaji, M. Nakabayashi, and S. Takano, “New explicit conditions of elliptic curve traces for FR-reduction,” IEICE Trans. Fundam. Electron., Commun., Comput. Sci., vol. E84-A, no. 5, pp. 1234–1243, 2001.

[23] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, “Secure network coding over the integers,” in Public Key Cryptography. Berlin, Germany: Springer-Verlag, 2010, pp. 142–160.

[24] S. Goldwasser, S. Micali, and R. L. Rivest, “A digital signature scheme secure against adaptive chosen-message attacks,” SIAM J. Comput., vol. 17, no. 2, pp. 281–308, 1988.

[25] P. S. L. M. Barreto and M. Naehrig, “Pairing-friendly elliptic curves of prime order,” in Selected Areas in Cryptography. Berlin, Germany: Springer-Verlag, 2006, pp. 319–331.

[26] Y. Deswarte, J.-J. Quisquater, and A. Saïdane, “Remote integrity checking,” in Integrity and Internal Control in Information Systems VI. Berlin, Germany: Springer-Verlag, 2004, pp. 1–11.

[27] D. L. G. Filho and P. S. L. M. Barreto, “Demonstrating data posses- sion and uncheatable data transfer,” Cryptology ePrint Archive, Tech. Rep. 2006/150, 2006.Online]. Available: http://eprint.iacr.org/

[28] 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.

[29] 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.

[30] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling public verifiability and data dynamics for storage security in cloud computing,” in Computer Security. Berlin, Germany: Springer-Verlag, 2009, pp. 355–370.

[31] S. G. Worku, C. Xu, J. Zhao, and X. He, “Secure and efficient privacy- preserving public auditing scheme for cloud storage,” Comput. Elect. Eng., vol. 40, no. 5, pp. 1703–1713, 2013.

[32] K. D. Bowers, A. Juels, and A. Oprea, “Proofs of retrievability: Theory and implementation,” in Proc. ACM Workshop Cloud Comput. Secur., 2009, pp. 43–54.

[33] Y. Dodis, S. Vadhan, and D. Wichs, “Proofs of retrievability via hardness amplification,” in Theory of Cryptography. Berlin, Germany: Springer-Verlag, 2009, pp. 109–127.