



A Distributed Storage Integrity Auditing for Secure Cloud Storage Services

Anuradha.R*, Dr. Y. Vijayalatha

Department of CSE & JNTUH
India

Abstract- Cloud storage provides a convenient means of storing and retrieval of huge amount of data. With this facility, organizations and individuals can outsource their data to cloud. Though cloud storage gives significant benefits to users, there are security concerns as well. Since the cloud users store their valuable business data in cloud, the security is an utmost concern. Moreover the cloud server has to be untrusted since it is accessed through a public network such as Internet. Many techniques came into existence to address the storage security problem in cloud. However, providing data audit with low communication and computation cost is important. Recently Wang et al. presented a distributed storage integrity auditing mechanism. This mechanism is based on security approaches such as erasure-coded data and homomorphic tokens. In this paper we practically implement the security mechanism using a prototype application that demonstrates the proof of concept. The empirical results revealed that the prototype is capable of preventing various attacks such as server colluding attacks, malicious data modification attack besides protecting byzantine failures.

Index Terms –Cloud computing, cloudstorage, data integrity, data dynamics, error localization

1. INTRODUCTION

In computing arena many trends came up. They include centralized computing, client server computing and distributed computing. Towards distributed computing the latest emerging technology is cloud computing. The increase in bandwidth availability made it possible to get new technologies. Cloud computing enables users to gain access to remote computing resources in pay per use fashion without investing for infrastructure. Many existing cloud service providers such as Amazon (Elastic Compute Cloud (EC2) [2]), Microsoft Azure etc. are providing cloud services. The users of cloud storage their valuable data in cloud. And they are at the mercy of the CSPs for the security of data [3], [4]. There are many incidents that provided that data loss is there in cloud storage [6], [7], [8]. One more problem is that users may not be able to maintain local copy of the data. When such data is lost, they will not be able to take it back. This is an important security consideration with regard to cloud data storage. There are some possibilities for fraudulent activities at CSP [10]. For instance, CSP may hide the storage inconsistencies in cloud [12], [13]; he may even remove some of the data of users to rent that space for monetary gains. Though these are assumptions, they are basis for the work in this paper.

In order to ensure that the data stored in cloud has integrity and availability, many methods came into existence. Many such methods depend on on-demand verification of data for its correctness. The verification of the cloud data has to be done without the full knowledge of such data. For achieving this some techniques were explored in [11], [12], [13]. Moreover they cloud data is not historical in nature. It does mean that it is subjected to frequent modifications. The data owners might update it constantly [14], [15], [16]. The operations on data include appending, modification, deletion and insertion. These dynamic features are to be integrated with security framework. When these operations are also protected, the cloud storage ensures the data consistency and availability. Many security models came into existence of late to ensure cloud data security. They include [11], [12], [13], [14], [19], [20], [21], [22]. These techniques are capable of security cloud storage. The problem with these techniques is that they focused on single server scenario. As explored in [23] quality of service is ensured but no guarantee is given for data availability. These techniques when applied directly can't work fine in distributed environment where multiple servers are involved. Later on some researchers proposed protocols that work in distributed environment [23], [24], [25]. These protocols ensure the data integrity in a multi-server environment besides supporting storage dynamics.

Recently Wang et al. [38] proposed flexible techniques that work in distributed environment. This technique uses erasure correcting code for data integrity. It also secures data from Byzantine servers [26]. This technique achieves security with less communication and computational overhead. The internal techniques used in the mechanism are erasure-coded data for distributed verification with homomorphic tokens. Error localization is another important aspect of the security mechanism. This will enable the mechanism to identify misbehaving servers. Token recomputation technique is also used to strike balance between data dynamics and error resilience. Third party auditing is used to have error-free storage and services.

In this paper we implemented the security mechanism proposed by Wang et al. [38] by developing an application to demonstrate proof of concept. The empirical results revealed that the security mechanism is robust to attacks. The remainder of this paper is structured into sections. Section II reviews relevant literature. Section III provides information about the proposed security mechanisms. Section IV gives details about the implementation. Section V presents experimental results while section VI concludes the paper.

2. RELATED WORK

Many security mechanisms came into existence as found in the literature. Proof of irretrievability [10] was proposed by Juels and Kaliski. It combines error correcting code and spot checking to ensure data integrity and irretrievability. On this model [17] is built and homomorphic authenticator based on the random linear function is used for secure cloud services. This resulted in secure mechanisms with less cost. An improved framework is proposed by Bowers et al. [18]. Later they also extended this to apply in distributed environment [23]. The commonality of this entire scheme is that they focused on static data. They depend on preprocessing for the success of the schemes. Their computational and communication cost is high. The reason for this is that every operation is done through error correcting code and random shuffling process. Of late Dodiset al. [20] studied the scheme to improve them. Provable Data Possession (PDP) was developed by Ateniese et al. [11] for untrusted storages. They used homomorphic tags which are based on public key cryptography for auditing the files. It proved to be computationally expensive as it takes more time to pre-compute tags. Later they improved the scheme for symmetric key-based cryptography [14]. This proved to have less overhead. However, the drawback of this scheme is that it worked only on single server. In case of server failure it could not provide guarantee for data. In two more recent studies [15] and [16] there was research on data dynamics. Merkle Hash Tree and BLS-based homomorphic authenticators are explored for supporting data dynamics. A skip-list based scheme was developed by Erway et al. [16] for provable data possession. Late ron Bellare et al. [36] used cryptography for storage security. However, this work comes under the traditional data integrity in which it is mandatory to maintain a local copy. Curtmola et al. [19] in their work used multiple servers and replicated data for ensuring data possession. This scheme covered multiple servers and ensured robustness of data. Another scheme known as P2P backup proposed by Lillibridge et al. [25] uses erasure codes for security. It also makes use of cryptographic keys. It can detect the data inconsistencies in cloud storage. RSA-based solution is provided by Filho and Barreto [37] for uncheatable data possession. The drawback of their approaches that it needs exponentiation over the entire data file. This is not practical in the real world. In [12] and [13] TPA (Third Party Auditing) was used to ensure data consistencies. Their scheme works for only encrypted files while auditors need to maintain the state for long time.

Schwarz and Miller [24] proposed a scheme that ensures static file integrity in cloud storage even when the storage is spread across multiple servers. In this paper some of the aspects of their mechanisms are used. Moreover our scheme supports data dynamics as well. The proposed scheme can identify storage inconsistencies and also misbehaving servers.

3. PROPOSED SECURITY MECHANISM

In the proposed security architecture, three parties are involved. They are cloud user, cloud service provider, and third party auditor. User is a person or organization who will outsource data to cloud. He involves in storage and data dynamics with the cloud service provider. The cloud service provider provides data storage space which can be used by cloud users. The operations are performed in an authenticated environment through Internet. Trusted Third Party is an expert in performing data verification tasks. He will perform data verification activities on behalf of cloud users. The design goals of the proposed security architecture include storage correctness, fast localization of data errors, dynamic data support, dependability, and light weight mechanisms. The adversary model is to capture all possible attack scenarios on the cloud storage. The algorithms used in this paper are taken from the work of Wang et al. [38]. The algorithms are briefly discussed here.

Algorithms Used

```
Algorithm 1. Token Precomputation.
1: procedure
2:   Choose parameters  $l, n$  and function  $f, \phi$ ;
3:   Choose the number  $t$  of tokens;
4:   Choose the number  $r$  of indices per verification;
5:   Generate master key  $K_{PRP}$  and challenge key  $k_{chal}$ ;
6:   for vector  $G^{(j)}, j \leftarrow 1, n$  do
7:     for round  $i \leftarrow 1, t$  do
8:       Derive  $\alpha_i = f_{k_{chal}}(i)$  and  $k_{prp}^{(i)}$  from  $K_{PRP}$ .
9:       Compute  $v_i^{(j)} = \sum_{q=1}^r \alpha_i^q * G^{(j)}[\phi_{k_{prp}^{(i)}}(q)]$ 
10:    end for
11:  end for
12:  Store all the  $v_i$ 's locally.
13: end procedure
```

Fig. 1 Token precomputation algorithm

As can be seen in fig. 1 it is evident that the token computation algorithm is able to perform computation of tokens for all blocks of data in vectors. These precomputed tokens can help in finding data correctness and also error localization in a later stage.

```

Algorithm 2. Correctness Verification and Error Localization.
1: procedure CHALLENGE(i)
2:   Recompute  $\alpha_i = f_{k_{chal}}(i)$  and  $k_{prp}^{(i)}$  from  $K_{PRP}$ ;
3:   Send  $\{\alpha_i, k_{prp}^{(i)}\}$  to all the cloud servers;
4:   Receive from servers:
       $\{R_i^{(j)} = \sum_{q=1}^r \alpha_i^q * G^{(j)}[\phi_{k_{prp}^{(i)}}(q)] | 1 \leq j \leq n\}$ 
5:   for ( $j \leftarrow m + 1, n$ ) do
6:      $R_i^{(j)} \leftarrow R_i^{(j)} - \sum_{q=1}^r f_{k_j}(s_{I_q, j}) \cdot \alpha_i^q, I_q = \phi_{k_{prp}^{(i)}}(q)$ 
7:   end for
8:   if ( $(R_i^{(1)}, \dots, R_i^{(m)}) \cdot \mathbf{P} == (R_i^{(m+1)}, \dots, R_i^{(n)})$ ) then
9:     Accept and ready for the next challenge.
10:  else
11:    for ( $j \leftarrow 1, n$ ) do
12:      if ( $R_i^{(j)} \neq v_i^{(j)}$ ) then
13:        return server j is misbehaving.
14:      end if
15:    end for
16:  end if
17: end procedure

```

Fig. 2- Algorithm for correctness verification

As can be seen in fig. 2, the algorithm computes correctness of given data blocks and also localized if any errors in the storage are found. The token based approach can help detect misbehaving users quickly. The proposed approach ensures that the server's misbehaving is tracked and presented. The cost of bandwidth required by this approach is very less when compared with other such approaches.

```

Algorithm 3. Error Recovery.
1: procedure
   % Assume the block corruptions have been detected
   among
   % the specified r rows;
   % Assume  $s \leq k$  servers have been identified
   misbehaving
2:   Download r rows of blocks from servers;
3:   Treat s servers as erasures and recover the blocks.
4:   Resend the recovered blocks to corresponding
      servers.
5: end procedure

```

Fig. 3 – Error recovery algorithm

As can be seen in fig. 3, the algorithm takes block corruptions' detection results and finds error recovery. This will be very useful to establish original data when it is damaged. That is the reason this algorithm is well designed in this paper. For more information about the algorithms and security mechanisms user is advised to read [38] where complete concept of security is presented.

4. EXPERIMENTAL RESULTS

The environment used to develop the proposed cloud security application is Java platform, a PC with 4 GB RAM, Core 2 Dual processor running in Windows 7 OS. Various experimental results are presented as series of graphs as given below.

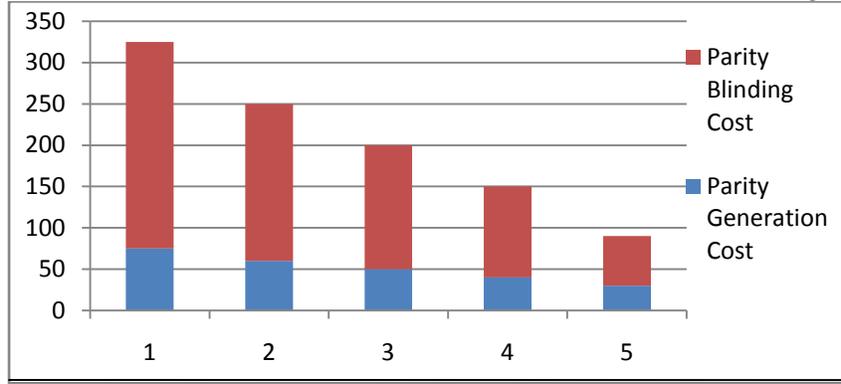


Fig. 4 – Performance comparison of two models

As can be seen in fig. 4, it is evident that the two parameter setting methods are compared for performance. The performance results between parity blinding cost and parity generation cost are presented in the graph.

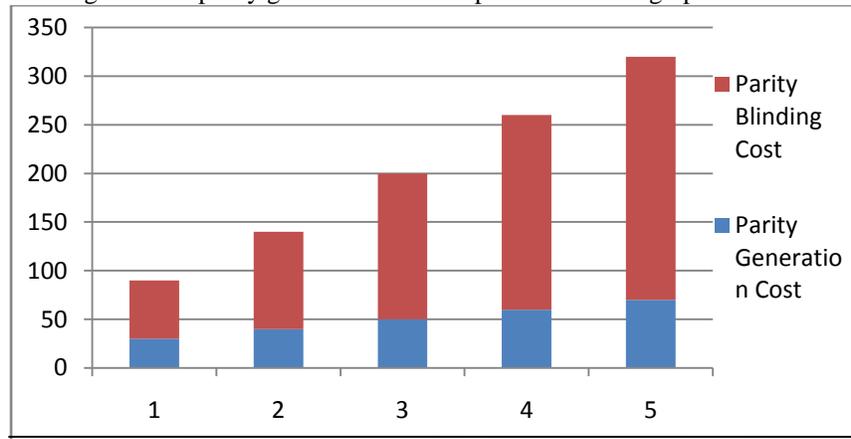


Fig. 5 – Performance comparison of two models

As can be seen in fig. 5, it is evident that the two parameter setting methods are compared for performance. The performance results between parity blinding cost and parity generation cost are presented in the graph.

5. CONCLUSION AND FUTUREWORK

In this paper we implemented the security mechanism proposed by Wang et al. [37]. The prototype application implemented by us demonstrates the proof of concept. We studied the data integrity problem and built security mechanisms that work in distributed environment. The flexible security scheme supports operations at block level. They include appending, deleting, updating. Erasure-correcting is used for guarantee of data dependability. Storage correctness and error localization are achieved by using homomorphic tokens and erasure codes. The error localization helps to find misbehaving server. Cloud users can delegate the integrity checks to third party auditor. Thus the security mechanism can prevent data modification attacks. The empirical results revealed the same. In future we propose a novel highly decentralized information accountability framework to keep track of the actual usage of the users' data in the cloud. We leverage the JAR programmable capabilities to both create a dynamic and traveling object, and to ensure that any access to users' data will trigger authentication and automated logging local to the JARs.

References

- [1] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring Data Storage Security in Cloud Computing," Proc. 17th Int'l Workshop Quality of Service (IWQoS '09), pp. 1-9, July 2009.
- [2] Amazon.com, "Amazon Web Services (AWS)," <http://aws.amazon.com>, 2009.
- [3] Sun Microsystems, Inc., "Building Customer Trust in Cloud Computing with Transparent Security," https://www.sun.com/offers/details/sun_transparency.xml, Nov. 2009.
- [4] K. Ren, C. Wang, and Q. Wang, "Security Challenges for the Public Cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69-73, 2012.
- [5] M. Arrington, "Gmail Disaster: Reports of Mass Email Deletions," <http://www.techcrunch.com/2006/12/28/gmail-disasterreportsof-mass-email-deletions>, Dec. 2006.

- [6] J. Kincaid, "MediaMax/TheLinkup Closes Its Doors," [http:// www.techcrunch.com/2008/07/10/mediamaxthelinkup-closesits- doors](http://www.techcrunch.com/2008/07/10/mediamaxthelinkup-closesits-doors), July 2008.
- [7] Amazon.com, "Amazon S3 Availability Event: July 20, 2008," <http://status.aws.amazon.com/s3-20080720.html>, July 2008.
- [8] S. Wilson, "Appengine Outage," [http://www.cio-weblog.com/ 50226711/appengine_outage.php](http://www.cio-weblog.com/50226711/appengine_outage.php), June 2008.
- [9] B. Krebs, "Payment Processor Breach May Be Largest Ever," http://voices.washingtonpost.com/securityfix/2009/01/payment_processor_breach_may_b.html, Jan. 2009.
- [10] A. Juels and B.S. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 584-597, Oct. 2007.
- [11] 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 (CCS\ '07), pp. 598-609, Oct. 2007.
- [12] M.A. Shah, M. Baker, J.C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," Proc. 11th USENIX Workshop Hot Topics in Operating Systems (HotOS '07), pp. 1-6, 2007.
- [13] M.A. Shah, R. Swaminathan, and M. Baker, "Privacy-Preserving Audit and Extraction of Digital Contents," Cryptology ePrint Archive, Report 2008/186, <http://eprint.iacr.org>, 2008.
- [14] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. Fourth Int'l Conf. Security and Privacy in Comm. Netowrks (SecureComm '08), pp. 1-10, 2008.
- [15] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing," Proc. 14th European Conf. Research in Computer Security (ESORICS '09), pp. 355-370, 2009.
- [16] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), pp. 213-222, 2009.
- [17] 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), pp. 90- 107, 2008.
- [18] K.D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Proc. ACM Workshop Cloud Computing Security (CCSW '09), pp. 43-54, 2009.
- [19] 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), pp. 411-420, 2008.
- [20] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of Retrievability via Hardness Amplification," Proc. Sixth Theory of Cryptography Conf. (TCC '09), Mar. 2009.
- [21] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, 2011.
- [22] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, preprint, 2012, doi:10.1109/TC.2011.245.
- [23] K.D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Proc. ACM Conf. Computer and Comm. Security (CCS '09), pp. 187-198, 2009.
- [24] T. Schwarz and E.L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS '06), pp. 12-12, 2006.
- [25] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A Cooperative Internet Backup Scheme," Proc. USENIX Ann. Technical Conf. (General Track), pp. 29-41, 2003.
- [26] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance and Proactive Recovery," ACM Trans. Computer Systems, vol. 20, no. 4, pp. 398-461, 2002.
- [27] L. Carter and M. Wegman, "Universal Hash Functions," J. Computer and System Sciences, vol. 18, no. 2, pp. 143-154, 1979.
- [28] J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasure-Coded Data," Proc. 26th ACM Symp. Principles of Distributed Computing, pp. 139-146, 2007.
- [29] J.S. Plank and Y. Ding, "Note: Correction to the 1997 Tutorial on Reed-Solomon Coding," Technical Report CS-03-504, Univ. of Tennessee, Apr. 2003.
- [30] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Storage Security in Cloud Computing," Proc. IEEE INFOCOM, Mar. 2010.
- [31] C. Wang, K. Ren, W. Lou, and J. Li, "Towards Publicly Auditable Secure Cloud Data Storage Services," IEEE Network Magazine, vol. 24, no. 4, pp. 19-24, July/Aug. 2010.
- [32] R.C. Merkle, "Protocols for Public Key Cryptosystems," Proc. IEEE Symp.Security and Privacy, 1980.
- [33] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," Proc. IEEE INFOCOM, Apr. 2009.
- [34] J.S. Plank, S. Simmerman, and C.D. Schuman, "Jerasure: A Library in C/C++ Facilitating Erasure Coding for Storage Applications - Version 1.2," Technical Report CS-08-627, Univ. of Tennessee, Aug. 2008.
- [35] M. Bellare, R. Canetti, and H. Krawczyk, "Keying Hash Functions for Message Authentication," Proc. 16th Ann. Int'l Cryptology Conf. Advances in Cryptology (Crypto '96), pp. 1-15, 1996.

- [36] M. Bellare, O. Goldreich, and S. Goldwasser, "Incremental Cryptography: The Case of Hashing and Signing," Proc. 14th Ann. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO '94), pp. 216-233, 1994.
- [37] D.L.G. Filho and P.S.L.M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Cryptology ePrint Archive, Report 2006/150, <http://eprint.iacr.org>, 2006.
- [38] Wang et al.

AUTHORS



R. Anuradha is student of GRIET College of Engineering and Technology, Hyderabad, AP, INDIA. She has received B.Tech Degree in computer science and engineering, pursuing M.Tech Degree in computer science and engineering. Her main research interest includes Cloud Computing, Databases and DWH.



Dr. Y. VIJAYALATA is an academician with more than 17 years of teaching and research experience received M.Tech.(Computer Science) degree from Birla Institute of Technology, Ranchi, India and Ph.D. from JNTUH, Kukatpally, Hyderabad. Working as a Professor in Department of Computer Science and Engineering at GokarajuRangaraju Institute of Engineering and Technology(GRIET), Hyderabad. She is also the Chair, WIE-AG IEEE Hyderabad Section(2012-2013,2013-2014). She was the Vice-Chair for IEEE WIE Affinity Group, Hyderabad Section(2011-2012). She is the Branch Counselor for IEEE Student Branch at GRIET.