Journal of Computer Applications ISSN: 0974 – 1925,
Volume-5, Issue EICA2012-4, February 10, 2012

Ensuring Data Integrity in Cloud Computing

Venkatesa Kumar V
Asst.Professor,
Dept of Computer Science and Engineering,
Anna University of Technology, Coimbatore,
Coimbatore, Tami Nadu
E-mail:mail2venkatesa@gmail.com

Poornima G
PG Scholar
Dept of Computer Science and Engineering,
Anna University of Technology, Coimbatore,
Coimbatore, Tami Nadu
E-mail: poornioct@gmail.com

Abstract— Cloud computing provides convenient on-demand network access to a shared pool of configurable computing resources. The resources can be rapidly deployed with great efficiency and minimal management overhead. Cloud is an insecure computing platform from the view point of the cloud users, the system must design mechanisms that not only protect sensitive information by enabling computations with encrypted data, but also protect users from malicious behaviours by enabling the validation of the computation result. In this paper, we propose a new data encoding scheme called layered interleaving, designed for time-sensitive packet recovery in the presence of bursty loss. It is high-speed data recovery scheme with minimal loss probability and using a forward error correction scheme to handle bursty loss. The proposed approach is highly efficient in recovering the singleton losses almost immediately and from bursty data losses.

Index Terms: *Cloud Computing, Data Integrity, Data Recovery, Layer Interleaving, Security.*

I. INTRODUCTION

Organisations today are increasingly looking towards Cloud Computing as a new revolutionary technology promising to cut the cost of development and maintenance and still achieve highly reliable and elastic services. The Cloud technology is a growing trend and is still undergoing lots of experiments. Cloud promises huge cost benefits, agility and scalability to the business [25]. All business data and software are stored on servers at a remote location referred to as Data centres. Data centre environment allows enterprises to run applications faster, with easier manageability and less maintenance effort, and more rapidly scale resources (e.g. servers, storage, and networking) to meet fluctuating business needs. A data center in cloud environment holds information that end-users would more traditionally have stored on their computers. This raise concerns regarding user privacy protection because users must outsource their data [2]. The movement of data to centralized services could affect the privacy and security of users' interactions with the files stored in cloud storage space. The use of virtualized infrastructure as a launching pad might introduce new attacks to user's data.

Data integrity is defined as the accuracy and consistency of stored data, in absence of any alteration to the data between two updates of a file or record. Cloud services should ensure data integrity and provide trust to the user privacy. Although outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, it's lacking of offering strong assurance of data integrity and availability may impede its wide adoption by both enterprise and individual cloud users [2]. Cloud computing poses privacy concerns primarily, because the service provider at any point in time, may access the data that is on the cloud. The Cloud service provider could accidentally or deliberately alter or delete some information from the cloud server. Hence, the system must have some sort of mechanism to ensure the data integrity. The current Cloud security model is based on the assumption that the user/customer should trust the provider. This is typically governed by a Service Level Agreement (SLA) that in general defines mutual provider and user expectations and obligations.

In order to ensure the integrity and availability of data in Cloud and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. However, the fact that users no longer have physical possession of data in the cloud prohibits the direct adoption of traditional cryptographic primitives for the purpose of data integrity protection [2]. Hence, the verification of cloud storage correctness must be conducted without explicit knowledge of the entire data files [2], [6], [12], [17]. The data stored in the cloud may not only be accessed but also be frequently updated by the users, including insertion, deletion, modification, appending, etc. Thus, it is also imperative to support the integration of this dynamic feature into the cloud storage correctness assurance, which makes the system design even more challenging [1]-[2]. In this paper, we analyse a method for encoding and data recovery in case of failures.

A.OUR CONTRIBUTION

The main contributions of this paper are

- The challenge-response protocol in our work further provides the localization of data error.
- We propose an efficient method for encoding the data to be transferred and stored in the Cloud.
- Finally, we propose an efficient data recovery method and performance analysis for the retrieval of lost data in Cloud.

B. ASSUMPTIONS

In this paper, we assume that the scheme supports secure and efficient dynamic operations on data blocks, including: update, delete and append.

The rest of the paper is organized as follows. Section II describes the related work. Section III introduces the system model and formulations. Then we provide the detailed description of our scheme in Section IV. Section V gives the security analysis. Section VI provides details on data recovery and Section VII on performance evaluations, Section VIII overviews the related work and concluding remark of the whole paper.

II. RELATED WORK

An effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users' data in the cloud was proposed by C. Wang, Q. Wang, K. Ren, and W. Lou in July 2009. C. Wang, Q. Wang, K. Ren, and W. Lou rely on erasure correcting code in the file distribution preparation to provide redundancies and guarantee the data dependability. This construction might drastically reduce the communication and storage overhead as compared to the traditional replication-based file distribution techniques. Their scheme achieves the storage correctness insurance as well as data error localization, that is, whenever data corruption has been detected during the storage correctness verification, their scheme can almost guarantee the simultaneous localization of data errors. Later in May 2011, Cong Wang, Qian Wang, Kui Ren, Wenjing Lou extended their work to allow user to audit the cloud storage with very lightweight communication and computation cost, proposed scheme that is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

A formal "Proof of Retrievability" (POR) model for ensuring the remote data integrity was described by A. Juels and J. Burton S. Kaliski in October 2007. Their scheme combines two methods spot-checking and error- correcting code to ensure both possession and retrievability of files on archive or backup service systems. H. Shacham and B. Waters in 2008 built on this model and constructed a random linear function based homomorphic authenticator which enables unlimited number of queries and requires less communication overhead[16]. An improved framework for POR protocols that generalizes both Juels and Shacham's work was illustrated [17]. All these schemes are focusing on static data. The effectiveness of their schemes rests primarily on the pre-processing steps that the user conducts before outsourcing the data file F. Any change to the contents of F, even few bits, must propagate through the error-correcting code, thus introducing significant computation and communication complexity was proposed by Bowers in 2009.

The "provable data possession" (PDP) model for ensuring possession of file on untrusted storages was defined by Ateniese et al [10]. Their scheme utilized public key based homomorphic tags for auditing the data file, thus providing public verifiability. However, their scheme requires sufficient computation overhead that can be expensive for an entire file. Later in their subsequent work during 2008, described a PDP scheme that uses only symmetric key cryptography. This method has lower-overhead than their previous scheme and allows for block updates, deletions and appends to the stored file, which has also been supported in our work. However, their scheme focuses on single server scenario and does not address small data corruptions, leaving both the distributed scenario and data error recovery issue unexplored.

A new efficient means of polynomial in the size of the input (i.e. key or data) was proposed by M. A. Shah, R. Swaminathan, and M. Baker during the year 2008 in "Privacy Preserving audit and extraction of digital contents". The main threat from the auditor is that it may glean important information from the auditing process that could compromise the privacy guarantees provided by the service. For example, even a few bits from a file containing medical history could reveal whether a customer has a disease. To ensure privacy, there exist different standards for the encrypted data and the encryption key. For the data, the system relies on (1) the strength of the encryption scheme and (2) the zero-knowledge property of the protocol for encryption-key audits.

To ensure file integrity across multiple distributed servers, using erasure-coding and block-level file integrity checks was proposed by T. S. J. Schwarz and E. L. Miller in 2009. However, their scheme only considers static data files. To verify data integrity using RSA-based hash for data possession in peer-to-peer file sharing networks was defined by D. L. G. Filho and P. S. L. M. Barreto in 2006. However, their proposal requires exponentiation over the entire data file, which is clearly impractical for the server whenever the file is large.

III. SYSTEM MODEL AND PROBLEM FORMULATION

The proposed system has three important entities,

User: Users store data in the cloud and depend on the cloud for all its computations on the data stored in the cloud server. User may be an individual or organization.

Cloud Service Provider (CSP): CSP contains resources and expertise in building and managing distributed cloud storage servers, owns and operates and leases the live Cloud computing systems.

Third Party Auditor (TPA): TPA has expertise and capabilities that users may not have, is trusted to assess, audit and expose risk of cloud storage services on behalf of the users upon request from the users.

A special entity is considered to ensure the security and dependability of the Cloud Server referred to as *Adversary Model*. The adversary is interested in continuously corrupting the user's data files stored on individual servers. Once a server is comprised, an adversary can pollute the original data files by modifying or introducing its own fraudulent data to prevent the original data from being retrieved by the user [2]. Proposed network architecture for cloud data storage is illustrated in Fig.1.

In cloud data storage system, users store their data in the cloud and no longer possess the data locally. Thus, the correctness and availability of the data files being stored on the distributed cloud servers must be guaranteed. One of the key issues is to effectively detect any unauthorized data modification and corruption, possibly due to server compromise.

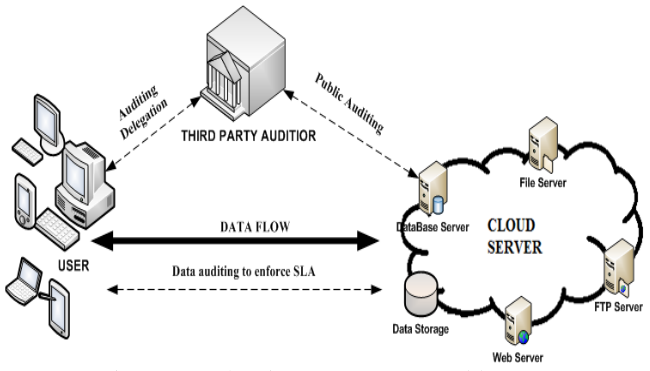

Figure. 1: Cloud Data Storage Architecture

Hence, it is presence of TPA is mandatory to assess, audit and expose risk of cloud storage services.In order to address these problems, our main scheme for ensuring cloud data storage is presented in this section. The first part of the section is devoted to a review of basic tools from coding theory that is needed in our scheme for file distribution across cloud servers. FEC encoders are typically parameterized with an (m , k) tuple . For each outgoing sequence of data packets, a total of (m+k) data and error correction packets are sent over the channel, resulting in an encoding overhead of k/m. The redundant information cannot be generated and sent until all data packets are available for sending. Consequently, the latency of packet recovery is determined by the rate at which the sender transmits data. Generating error correction packets from less than data packets at the sender is not a viable option—even though the data rate in this channel is low, the receiver and/or network could be operating at near full capacity with data from other senders and FEC is susceptible to bursty losses.

IV. DETAILED DESCRIPTION

The detailed description covers the important modules of the paper.

A.FILE DISTRIBUTION PREPARATION

The erasure-correcting code may be used to tolerate multiple failures in distributed storage systems. In cloud data storage, we rely on this technique to disperse the data file F redundantly across a set of d distributed servers. The layer interleaving technique is used to determine the c redundancy parity vectors from r data vectors in such a way that the original r data vectors can be reconstructed from any r out of the r + c data and parity vectors. By placing each of the r + c vectors on a different server, the original data file can survive the failure of any c of the r + c servers without any data loss, with a space overhead of c/r. The unmodified r data file vectors together with c parity vectors are distributed across r + c different servers.

The user obtains the encoded file by multiplying **F** by **A**

that is , $\mathbf{G} = \mathbf{F} \cdot \mathbf{A} = (G(1),G(2), \ldots , G(m),G(m+1),. , G(n))$
$= (F1, F2, \ldots , Fm, G(m+1), \ldots , G(n))$,

where F is the actual file and A is derived from a Vandermonde matrix, is a matrix with the terms of a geometric progression in each row. For a interleave index of 3, the first block containing data packets numbered (0,3,6,...(r-1).c), the second with data packets numbered (1,4,7,..,((r-1).c)+1) and the third with data packets numbered (2,5,8,...((r-1).c)+2).

B.TPA IMPLEMENTATION

User can delegate the task of auditing to an independent third party auditor, making the cloud storage publicly verifiable. For an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy. Namely, TPA should not learn user's data content through the delegated data auditing. The proposed scheme can support privacy-preserving third party auditing [20]. TPA auditing should

- Introduce no new vulnerabilities.
- Protect customer data privacy.
- Protect proprietary provider information.
- Audit results must be trustworthy.
- Avoid prescribing a technology.

when user delegates the auditing responsibility to the TPA, he sends all the attributes required for verifying the Cloud server in a secured encrypted manner. The TPA verifies the cloud server for the user and shares the correctness guarantee results for the cloud server for which the user requested verification.

V. SECURITY ANALYSIS

To achieve assurance of data storage correctness and data error localization simultaneously, this paper focuses on a scheme Challenge-Response protocol.

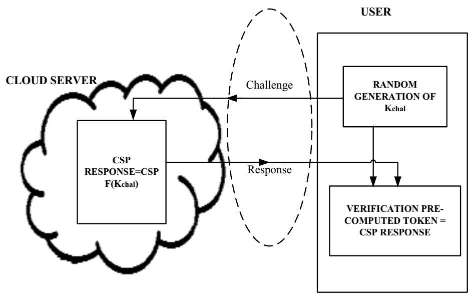
USER
CLOUD SERVER
Challenge
RANDOM GENERATION OF Kchal
CSP RESPONSE=CSP F(Kchal)
Response
VERIFICATION PRE-COMPUTED TOKEN = CSP RESPONSE

Figure 2: Challenge Response Protocol

A. Challenge Token Creation

The main idea is - when a file is distributed to the cloud, the user pre-computes a certain number of short verification tokens on individual vector $G^{(j)}$ ($j \in \{1, \ldots, n\}$), each token covering a random subset of data blocks that would be distributed to the different cloud servers. Later, when the user wants to make sure the storage correctness for the data in the cloud, he challenges the cloud servers with a set of randomly generated block indices. Upon receiving challenge, each cloud server computes a short "signature" over the specified blocks and returns them to the user. The values of these signatures should match the corresponding tokens pre-computed by the user. Suppose if the user wants to challenge the cloud server t times to ensure the correctness of data storage, the user must pre-compute x verification tokens for each $G^{(j)}$ ($j \in \{1, \ldots, n\}$), a challenge key k_{chal} and a master permutation key K_{PRP}. To generate the i^{th} token for server j, the user acts as follows,

1. Derive a random challenge value α_i and a permutation key $k^{(i)}_{prp}$ based on K_{PRP}.

2. Compute the set of r randomly-chosen indices.

3. Calculate the token $v^{(j)}_{i}$ using the random challenge value α_i.

After token generation, the user has the choice of either keeping the pre-computed tokens locally or storing them in encrypted form on the cloud servers.

B. Correctness Verification

The response values from servers for each challenge not only determine the correctness of the distributed storage, but also contain information to locate potential data error(s).
The procedure of the i^{th} challenge-response for verification over the d servers is described as follows:

- The user reveals the permutation key to each server.
- The server storing vector $G^{(j)}$ aggregates those k rows specified by index permutation key into a linear combination.
- Upon receiving linear combination from all the servers, the user takes away blind values.
- Then the user verifies whether the received values remain a valid codeword determined by secret matrix **P.**

VI. DATA RECOVERY

FEC information is usually added to mass storage devices to enable recovery of corrupted data. The redundancy allows the receiver to detect a limited number of errors that may occur anywhere in the message, and often to correct these errors without retransmission. The two challenges in using FEC are

•Rate sensitivity.
•Burst susceptibility.

Interleaving is a standard encoding technique used to combat bursty loss, where error correction packets are generated from alternate disjoint blocks of data rather than from consecutive packets. For example, with an interleave index of 4, the encoder would create correction packets separately from three disjoint blocks. The first block containing data packets numbered (0,4,,...(m-1).4), the second with data packets numbered (1,5,7,..,((m-1).4)+1), the third with data packets numbered (2,6,8,...((m-1).4)+2) and the fourth block would contain (3,7,…((m-1).4)+3) Interleaving adds burst tolerance to FEC, but exacerbates its sensitivity to sending rate. With an interleave index of i and an encoding rate of (m, k), the sender would have to wait i.(m-1)+1 for packets before sending any redundancy information.

VII. PERFORMANCE EVALUATION

The performance of the system is evaluated based on the following parameters.

- Time required for generation of tokens.
- Cost of file distribution.
- Correctness Verification efficiency.
- Recovery Latency

Figure. 3. Illustrates the difference between a traditional FEC code and layered interleaving by plotting a 50-element moving average of recovery latencies for both codes. Our experiment is conducted on a system with an Intel Core 2 processor running at 2.4 GHz, 2 GB RAM, and a 320 GB Serial Hard disk drive. The results represent the mean of all trials. The channel is configured to lose singleton packets randomly at a loss rate of 0.1% and additionally lose long bursts of 30 packets at occasional intervals. Both codes are configured with and recover all lost packets.

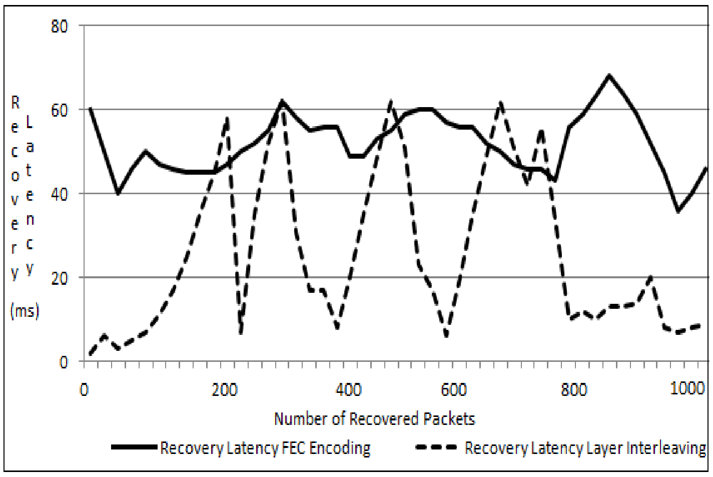

Figure 3: FEC Encoding and Layer Interleaving

FEC encoding, particularly Reed–Solomon uses interleave of 20, and layered interleaving uses interleaves of (1, 40). Reed Solomon, Layer Interleaving - both have a maximum tolerable burst length of 40 packets. The Reed–Solomon code recovers all lost packets with roughly the same latency, whereas layered interleaving recovers singleton losses almost immediately and exhibits latency spikes whenever the longer loss burst occurs.

VIII. CONCLUSION AND FUTURE WORK

Cloud Computing is gaining remarkable popularity in the recent years for its benefits in terms of flexibility, scalability, reliability and cost effectiveness. Despite all the promises however, Cloud Computing has one problem: Security. In this paper, we studied the problems of data security in cloud data storage, which is essentially a distributed storage system. An effective and flexible distributed scheme is proposed to ensure the correctness of users' data in the cloud servers. If this correctness verification is too much resource consuming on the user's side, the task can be delegated to the third party auditor and the pre-computed tokens could be either in the user's local device or cloud server in encrypted format. By detailed security and performance analysis, we show that our scheme is highly efficient in recovering the singleton losses almost immediately and recovers from bursty data losses. We envisage several possible directions for future research on this area.

As our future work we focus on reducing the impact in maintaining the challenge key in user's local space. For this we can split the challenge key into several parts- partial keys and maintain those keys in different cloud server and yet ensure security and data transparency. This might reduce the space overhead and possible cross verification of the verification process of a TPA by other TPAs.

REFERENCES

1. C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in Proc. of IWQoS'09, July 2009, pp. 1–9.
2. Cong Wang, Qian Wang, Kui Ren, Wenjing Lou, "Towards Secure and Dependable Storage Services in Cloud Computing," IEEE transactions on Services Computing, 06 May 2011.
3. A. Juels and J. Burton S. Kaliski, "PoRs: Proofs of retrievability for large files," in Proc. of CCS'07, Alexandria, VA, October 2007, pp. 584–597.
4. Sun Microsystems, Inc., "Building customer trust in cloud computing with transparent security," Online at https://www.sun. Com/offers/details/sun transparency.xml, November 2009.
5. M. Arrington, "Gmail disaster: Reports of mass email deletions," online at http: // www. techcrunch.com/2006/12/ 28/ gmail-disasterreports-of-mass-email-deletions/, December 2006.
6. D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Cryptology ePrint Archive, Report 2006/150, 2006, http://eprint.iacr.org/.
7. T. S. J. Schwarz and E. L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," Proc. of ICDCS '06, pp. 12–12, 2006.
8. B. Krebs, "Payment Processor Breach May Be Largest Ever," Online at http://voices.washingtonpost.com/securityfix/2009/01/ payment processor breach may b.html, Jan. 2009.
9. Amazon.com, "Amazon web services (aws)," Online at http:// aws.amazon.com/, 2009
10. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. of CCS'07, Alexandria, VA, October 2007, pp. 598–609.
11. M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to keep online storage services honest," in Proc. of HotOS'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 1–6.
12. M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008, http://eprint.iacr.org/.
13. G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. of SecureComm'08, 2008, pp. 1–10.
14. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in Proc. of ESORICS'09, volume 5789 of LNCS. Springer- Verlag, Sep. 2009, pp. 355–370.
15. C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. of CCS'09, 2009, pp. 213–222.
16. H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. of Asiacrypt'08, volume 5350 of LNCS, 2008, pp. 90–107.
17. K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: Theory and implementation," in Proc. of ACM workshop on Cloud Computing security (CCSW'09), 2009, pp. 43–54.
18. R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-replica provable data possession," in Proc. of ICDCS'08. IEEE Computer Society, 2008, pp. 411–420.
19. John W.Rittinghouse and James F.Ransome, "Cloud Computing Implementation, Management and Security," in CRC Press 2010 by Taylor and Francis Group.
20. M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to keep online storage services honest," in Proc. of HotOS'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 1–6.
21. M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008, http://eprint.iacr.org/.
22. G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. of SecureComm'08, 2008, pp. 1–10.

23. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling public verifiability and data dynamics for storage security in cloud computing,” in Proc. of ESORICS’09, volume 5789 of LNCS. Springer- Verlag, Sep. 2009, pp. 355–370.
24. Amazon.com, “Amazon s3 availability event: July 20, 2008,” Online at http://status.aws.amazon.com/s3-20080720.html, July 2008.
25. Prashant Srivastava, Satyam Singh, Ashwin Alfred Pinto, Shvetank Verma, Vijay K. Chaurasiya, Rahul Gupta, "An architecture based on proactive model for security in cloud computing" in IEEE-International Conference on Recent Trends in Information Technology, June 2011.
26. M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, “A cooperative internet backup scheme,” in Proc. of the 2003 USENIX Annual Technical Conference (General Track), 2003, pp. 29–41.
27. M. Castro and B. Liskov, “Practical byzantine fault tolerance and proactive recovery,” ACM Transaction on Computer Systems, vol. 20, no. 4, pp. 398–461, 2002.
28. 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, 2010.
29. R. C. Merkle, “Protocols for public key cryptosystems,” in Proc. of IEEE Symposium on Security and Privacy, Los Alamitos, CA, USA, 1980.
30. Q. Wang, K. Ren, W. Lou, and Y. Zhang, “Dependable and secure sensor data storage with dynamic integrity assurance,” in Proc. of IEEE INFOCOM’09, Rio de Janeiro, Brazil, April 2009.
31. M. Bellare, O. Goldreich, and S. Goldwasser, “Incremental cryptography: The case of hashing and signing,” in Proc. of CRYPTO’94, volume 839 of LNCS. Springer-Verlag, 1994, pp. 216– 233.

Mr.V.Venkatesakumar is currently working as an Assistant Professor in Anna University of Technology Coimbatore. He received Bachelor and Master Degree, B.E and M.E from Anna University Chennai. He has 6 years of Teaching Experience. Currently he is an academician and doing his doctorate. His research area includes Operating System, Software Engineering and Web Technologies.

Ms.Poornima G has received the B.E. degree from Anna University, Chennai in 2007. She is currently pursuing her M.E. degree in the Computer Science and Engineering Department at Anna University of Technology, Coimbatore. She has twenty eight months of industrial experience in an IT company. Her research interests include Software Engineering, Operating Systems and Cloud Computing.