ISSN:0975-9646

A Novel Third Party Auditability and Dynamic Based Security in Cloud Computing

J.Vinitha Mary*, R.Backiyalakshmi#

*M.Tech Student Department of Computer Science and Engineering,
PRIST University Pondicherry, India.
#Assistant professor Department of Computer Science and Engineering,
PRIST University Pondicherry, India.

Abstract - **Cloud computing refers to the use and access of multiple server-based computational resources via a digital network (WAN, Internet connection using the World Wide Web, etc.). Cloud users may access the server resources using a computer or other device. In cloud computing, applications are provided and managed by the cloud server and data is also stored remotely in the cloud configuration. Users do not download and install applications on their own device or computer; all processing and storage is maintained by the cloud server. Security is the major problem in Cloud as its open to many users. In order to maintain the integrity of data in cloud I am providing public audit ability to the network. This can be done by third party auditor (TPA) on behalf of the cloud client to verify the integrity of the dynamic data stored in the cloud. This will eliminates the interference of Client to check their intactness which can be important in achieving economies of scale for Cloud Computing. The forms of data operation such as block modification, insertion, and deletion is also a significant in the cloud function which will be done here. I detect the security problems of direct extensions with dynamic data updates from prior works and then show how to construct an elegant verification scheme for the seamless integration of these two salient features in my protocol design. In particular, to achieve efficient data dynamics, I improve the existing proof of storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks, I further explore the technique of bilinear aggregate signature to extend my main result into a multi-user setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance analysis show that the proposed schemes are highly efficient and provably secure**

1. Introduction

Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the “software as a service” (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. Meanwhile, the increasing network bandwidth and reliable yet flexible network connections make it even possible that clients can now subscribe high-quality services from data and software that reside solely on remote data centers. One of the biggest concerns with cloud data storage is that of data integrity verification at untrusted servers. For example, the storage service provider, which experiences Byzantine failures occasionally, may decide to hide the data errors from the clients for the benefit of their own. What is more serious is that for saving money and storage space the service provider might neglect to keep or deliberately delete rarely accessed data files which belong to an ordinary client. Consider the large size of the outsourced electronic data and the client’s constrained resource capability, the core of the problem can be generalized as how can the client find an efficient way to perform periodical integrity verifications without the local copy of data files In all these works, great efforts are made to design solutions that meet various requirements: high scheme efficiency, stateless verification, unbounded use of queries and irretrievability of data, etc.

Considering the role of the verifier in the model, all the schemes presented before fall into two categories: private audit ability and public audit ability. Although schemes with private audit ability can achieve higher scheme efficiency, public audit ability allows anyone, not just the client (data owner), to challenge the cloud server for correctness of data storage while keeping no private information. Then, clients are able to delegate the evaluation of the service performance to an independent third party auditor (TPA), without devotion of their computation resources. In the cloud, the clients themselves are unreliable or may not be able to afford the overhead of performing frequent integrity checks. Thus, for practical +use, it seems more rational to equip the verification protocol with public auditability, which is expected to play a more important role in achieving economies of scale for Cloud Computing.

Moreover, for efficiency consideration, the outsourced data themselves should not be required by the verifier for the verification purpose. Another major concern among previous designs is that of supporting dynamic data operation for cloud data storage applications. In Cloud Computing, the remotely stored electronic data might not only be accessed but also updated by the clients, e.g., through block modification,deletion, insertion, etc. Unfortunately, the state of the art in the contextof remote data storage mainly focus on static data files and the importance of this dynamic data updates has received limited attention Moreover, as will be shown later, the direct extension of the current provable data possession (PDP) schemes to support data dynamics may lead to security loopholes. Although there are many difficulties faced by researchers, it is well believed that supporting dynamic data operation can be of vital importance to the practical application of storage outsourcing services. In view of the key role of public audit ability and data dynamics for cloud data storage, An efficient construction for the seamless integration of these two components in the protocol design is proposed in the issue. My contribution can be summarized as follows:

1. To motivate the public auditing system of data storage security in Cloud Computing, and propose a protocol supporting for fully dynamic data operations, especially to support block insertion, which is missing in most existing schemes.
2. To extend my scheme to support scalable and efficient public auditing in Cloud Computing. In particular, my scheme achieves batch auditing where multiple delegated auditing tasks from different users can be performed simultaneously by the TPA.
3. To prove the security of my proposed construction and justify the performance of my scheme through concrete implementation and comparisons with the state of the art.

Related Work:

Recently, much of growing interest has been pursued in the context of remotely stored data verification. The first to consider public audit ability in their defined "provable data possession" model for ensuring possession of files on untrusted storages. In their scheme, they utilize RSA-based homomorphism tags for auditing outsourced data, thus public audit ability is achieved. However, Atenieseetal. Does not consider the case of dynamic data storage, and the direct extension of their scheme from static data storage to dynamic case may suffer design and security problems. A dynamic version of the prior PDP scheme proposed. However, the system imposes a priori bound on the number of queries and does not support fully dynamic data operations, i.e., it only allows very basic block operations with limited functionality, and block insertions cannot be supported.

Consider dynamic data storage in a distributed scenario, and the proposed challenge-response protocol can both determine the data correctness and locate possible errors. They only consider partial support for dynamic data operation. "Proof of irretrievability" model, where spot-checking and error-correcting codes are used to ensure both "possession" and "irretrievability" of data files on archive service systems. Specifically, some special blocks called "sentinels" are randomly embedded into the data file F for detection purpose, and F is further encrypted to protect the positions of these special blocks. The number of queries a client can perform is also a fixed priori, and the introduction of precompiled "sentinels" prevents the development of realizing dynamic data updates. In addition, public audit ability is not supported in their scheme. Design an improved PoR scheme with full proofs of security in the security model. They use publicly verifiable homomorphism authenticators built from BLS signatures, based on which the proofs can be aggregated into a small authenticator value, and public irretrievability is achieved. Still, the authors only consider static data files, were the first to explore constructions for dynamic provable data possession. They extend the PDP model to support provable updates to stored data files using rank-based authenticated skip lists. This scheme is essentially a fully dynamic version of the PDP solution. To support updates, especially for block insertion, they eliminate the index information in the "tag" computation in PDP model and employ authenticated skip list data structure to authenticate the tag information of challenged or updated blocks first before the verification procedure. However, the efficiency of their scheme remains unclear. Although the existing schemes aim at providing integrity verification for different data storage systems, the problem of supporting both public audit ability and data dynamics has not been fully addressed. How to achieve a secure and efficient design to seamlessly integrate these two important components for data storage service remains an open challenging task in Cloud Computing. Before the introduction of my proposed construction, To present two basic solutions (i.e., the MAC-based and signature-based schemes) for realizing data audit ability and discuss their demerits in supporting public audit ability and data dynamics.

Second, To generalize the support of data dynamics to both PoR and PDP models and discuss the impact of dynamic data operations on the overall system efficiency both. In particular, To emphasize that while dynamic data updates can be performed efficiently in PDP models more efficient protocols need to be designed for the update of the encoded files in PoR models. For completeness, the designs for distributed data storage security are also discussed in Section 3.5.

Third, in Section 3.4, to extend my data auditing scheme for the single client and explicitly include a concrete description of the multi-line data auditing scheme.I also redo the whole experiments and present the performance comparison between the multi client data auditing scheme and the individual auditing scheme Organization. The rest of the paper is organized as follows:

In Section 2, To define the system model, security model, and my design goals. In further analyze the experiment results and show the practicality of cloud data storage architecture.

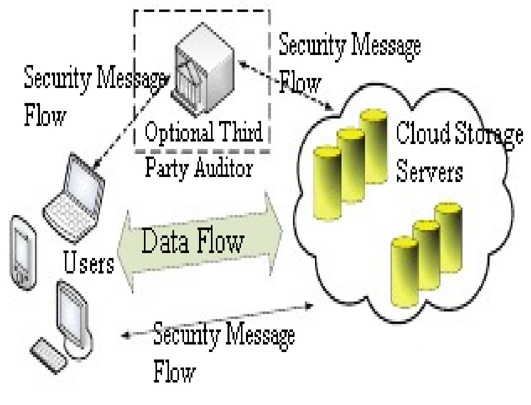

Fig.1.1 Cloud data storage architecture

2. Problem Statements

2.1 System Model:

Representative network architecture for cloud data storage is illustrated in Fig. 1. Three different network entities can be identified as follows:

I. **Client:** an entity, which has large data files to be stored in the cloud and relies on the cloud for data maintenance and computation, can be either individual consumers or organizations.
II. **Cloud Storage Server (CSS):** an entity, which is managed by Cloud Service Provider (CSP), has significant storage space and computation resource to maintain the clients' data.
III. **Third Party Auditor:** an entity, which has expertise and capabilities that clients do not have, is trusted to assess and expose risk of cloud storage services on behalf of the clients upon request.

In the cloud paradigm, by putting the large data files on the remote servers, the clients can be relieved of the burden of storage and computation.. That is, clients should be equipped with certain security means so that they can periodically verify the correctness of the remote data even without the existence of local copies. In case that client does not necessarily have the time, feasibility or resources to monitor their data, they can delegate the monitoring task to a trusted TPA.
In this paper, only verification schemes with public audit ability is considered : any TPA in possession of the public key can act as a verifier. To assume that TPA is unbiased while the server is untrusted. For application purposes, the clients may interact with the cloud servers via CSP to access or retrieve their presorted data. More importantly, in practical scenarios, the client may frequently perform block-level operations on the data files.
The most general forms of these operations considerd in this paper are modification, insertion, and deletion. Note that don't address the issue of data privacy in this paper, as the topic of data privacy in Cloud Computing is orthogonal to the problem studied here.

2.2 Security Model:
Following the security model there exists no polynomial time algorithm that can cheat the verifier with no negligible probability; and 2) there exists a polynomial time extractor that can recover the original data files by carrying out multiple challenges-responses. The client or TPA can periodically challenge the storage server to ensure the correctness of the cloud data, and the original files can be recovered by interacting with the server.
The scheme is correct if the verification algorithm accepts when interacting with the valid prover (e.g., the server returns a valid response) and it is sound if any cheating server that convinces the client it is storing the data file is actually storing that file.

2.3 Design Goals:
My design goals can be summarized as the following:
1. Public audit ability for storage correctness assurance: to allow anyone, not just the clients who originally stored the file on cloud servers, to have the capability to verify the correctness of the stored data on demand.
2. Dynamic data operation support: to allow the clients to perform block-level operations on the data files while maintaining the same level of data correctness assurance. The design should be as efficient as possible so as to ensure the seamless integration of public audit ability and dynamic data operation support.
3. Block less verification: no challenged file blocks should be retrieved by the verifier (e.g., TPA) during verification process for efficiency concern.

3. THE PROPOSED SCHEME
In this section, security protocols for cloud data storage service is provided with the aforementioned research goals in mind. To start with some basic solutions aiming to provide integrity assurance of the cloud data and discuss their demerits. Then, to present my protocol which supports public audit ability and data dynamics. To also show how to extent my main scheme to support batch auditing for TPA upon delegations from multi user. Assume the outsourced data file F consists of a finite ordered set of blocks m1; m2; . . . mn. One straightforward way to ensure the data integrity is to pre-compute MACs for the entire data file. Specifically, before data outsourcing, the data owner pre-computes MACs of F with a set of secret keys and stores them locally. During the auditing process, the data owner each time reveals a secret key to the cloud server and asks for a fresh keyed MAC for verification.

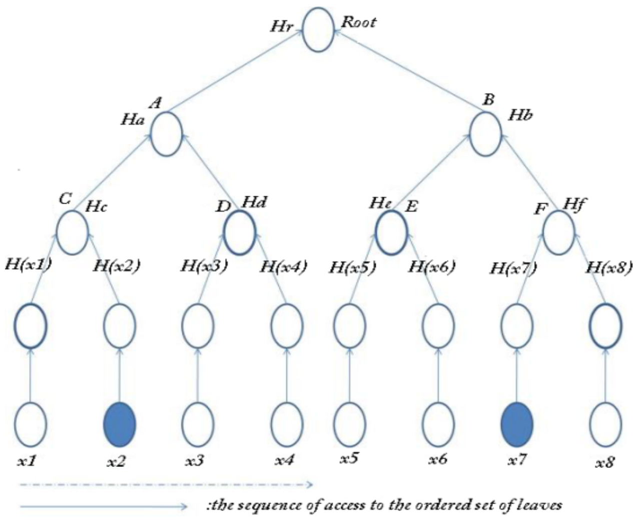

Fig.3.1. Merkle Hash Tree authentication of data elements.I treat the leaf nodes h(x1),…,h(xn) as the left to right sequence.

This approach provides deterministic data integrity assurance straightforwardly as the verification covers all the data blocks. However, the number of verifications allowed to be performed in this solution is limited by the number of secret keys. Once the keys are exhausted, the data owner has to retrieve the entire file of F from the server in order to compute new MACs, which is usually impractical due to the huge communication overhead. Moreover, public Audit ability is not supported as the private keys are required for verification. Another basic solution is to use signatures communication overhead and greatly affects system efficiency. Notice that the above solutions can only support the case of static data and none of them can deal with dynamic data updates.

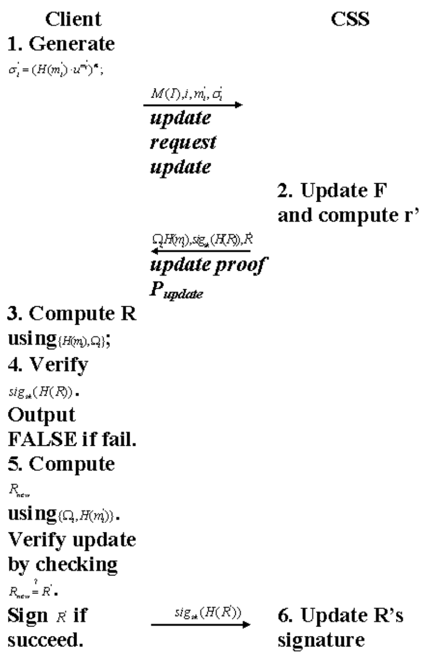
Client
CSS
1. Generate
$\sigma_i' = (H(m_i') \cdot u^{m_i'})^{\alpha}$;
$M(T), i, m_i', \sigma_i'$
update request update
2. Update F and compute r'
$\Omega_i, H(m_i), sig_{sk}(H(R)), R'$
update proof P_{update}
3. Compute R using $\{H(m_i), \Omega_i\}$;
4. Verify $sig_{sk}(H(R))$. Output FALSE if fail.
5. Compute R_{new} using $(\Omega_i, H(m_i'))$. Verify update by checking $R_{new} \overset{?}{=} R'$.
Sign R' if succeed.
$sig_{sk}(H(R'))$
6. Update R's signature

Fig.3.2. Protocols for Default Integrity Verification

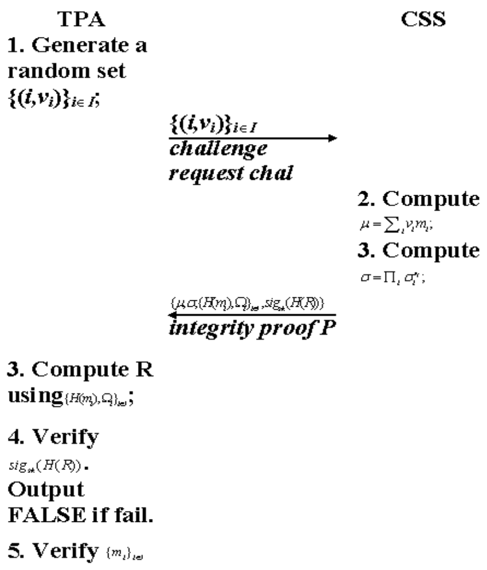
TPA
CSS
1. Generate a random set $\{(i,v_i)\}_{i\in I}$;
$\{(i,v_i)\}_{i\in I}$
challenge request chal
2. Compute $\mu = \sum_i v_i m_i$;
3. Compute $\sigma = \prod_i \sigma_i^{v_i}$;
$\{\mu, \sigma, \{H(m_i), \Omega_i\}_{i\in I}, sig_{sk}(H(R))\}$
integrity proof P
3. Compute R using $\{H(m_i), \Omega_i\}_{i\in I}$;
4. Verify $sig_{sk}(H(R))$. Output FALSE if fail.
5. Verify $\{m_i\}_{i\in I}$

Fig.3.4. The Protocol for Provable Data Update (Modification and Insertion)

4. Conclusion

To ensure cloud data storage security, it is critical to enable a TPA to evaluate the service quality from an objective and independent perspective. Public audit ability also allows clients to delegate the integrity verification tasks to TPA while they themselves can be unreliable or not be able to commit necessary computation resources performing continuous verifications. Another major concern is how to construct verification protocols that can accommodate Dynamic data files. In this paper, theexplored problems of providing simultaneous public audit ability and data dynamics for remote data integrity check in Cloud Computing. My construction is deliberately designed to meet these two important goals while efficiency being kept closely in mind. To achieve efficient data dynamics, To improve the existing proof of storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks,To further explore the technique of bilinear aggregate signature to extend my main result into a multi-user setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance analysis show that the proposed scheme is highly efficient and provably secure.

5. References

[1] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," *Proc. of IEEE INFOCOM*, 2009.

[2] Qian Wang, Cong Wang, KuiRen, Wenjing Lou, Jin Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE transactions on vol.:22, issue: 5, 2011.

[3] Cong Wang; Qian Wang; KuiRen; Wenjing Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing, *Proc. of IEEE INFOCOM*, 2010.

[4] Cong Wang; Qian Wang; KuiRen; Ning Cao; Wenjing Lou, "Towards Secure and Dependable Storage Services in Cloud Computing," IEEE transactions on vol.:5, issue: 2, 2012.

[5] Singh, Y.; Kandah, F.; Weiyi Zhang, "A Secured Cost-effective Multi-Cloud Storage in Cloud Computing," IEEE conference on (INFOCOM WKSHPS), 2011.

[6] 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 Symp. Research in Computer Security (ESORICS '09), pp. 355-370, 2009.

[7] Twenty Experts Define Cloud Computing, http://cloudcomputing.syscon.com/read/612375_p.htm [18 July 2008]

D. Hamilton. 'Cloud computing' seen as next wave for technology investors. *Financial Post*, 04 June 2008. http://www.financialpost.com/money/story.html

[8] Mike Ricciuti, "Stallman: Cloud computing is 'stupidity'", http://news.cnet.com/8301-1001_3-10054253-92.html

[9] K. Keahey, I. Foster, T. Freeman, and X. Zhang. Virtual workspaces: Achieving quality of service and quality of life in the Grid.*Scientific Programming*, 13(4):265-275, October 2005.

[10] I. Llorente, OpenNebula Project. http://www.opennebula.org/ [23 July 2008]

[11] Amazon Elastic Compute Cloud (EC2), http://www.amazon.com/ec2/ [18 July 2008]

Author Profile

Ms.J.Vinitha Mary, Received The B.E(Computer Science Engineering) From I.F.E.T College of Engg, Anna University, Chennai, Tamil Nadu, India and Presently Pursuing Final Year M.TECH CSE, In PRIST University, Puducherry Campus, Puducherry, India in 2012 and 2014.

Ms.R.Backiyalakshmi ,Received The M.Tech In Computer Science And Engineering. Presently she is a Working Assistant Professor in Computer Science and Engineering at PRIST University, Puducherry Campus, and Puducherry, India.