ISSN:0975-9646

A Privacy Preserving and Data Dynamics for Storage Security in Cloud Computing by using Third Party Auditor

M.Gayathri[#1], S.Kanchana[#2], R.Rajkumar[#3], Dr.S.Rajkumar[#4]

1. Assistant Professor, Computer Science and Engineering,
Dhanalakshmi Srinivasan college of Engineering, Perambalur – 621212, Tamilnadu, India.
2. Associate Professor, Computer Science and Engineering,
Indra Ganesan College of Engineering, Tiruchirappalli- 620012, Tamilnadu, India.
3. Senior Assistant Professor, Department of ComputerScience Engineering,
Chettinad College of Engineering &Technology, Karur-639114, Tamilnadu, India.
4. Assistant Professor, School of Mechanical and Electromechanical Engineering,
Hawassa Institute of Technology, Hawassa University. Hawassa, Ethiopia.

Abstract- **Privacy preserving is a crucial technology in cloud computing. The task of allowing Third Party Auditor, verify the integrity of the dynamic data stored in the cloud. The Third party auditor can eliminate the direct interaction between the client and the cloud server. Our goal can be focus on providing data dynamics and privacy preserving. To achieve the efficient data dynamics by using the Classic Markel Hash Tree construction for block tag authentication. Bilinear aggregate signature to perform the multiple auditing tasks. The proposed scheme does not leak any private information. After that, through theoretical analysis and experimental results, we demonstrate that the proposed scheme has a good performance and highly efficient.**

Keywords—Data Storage, Data dynamics, Privacy Preserving, cloud computing.

I. INTRODUCTION

Cloud computing is an internet based development and useof computer technology. “Cloud” brings many security challenges. Data integrity verification at untrusted server in the cloud data storage is one of the major concerns. An increasing number of clients store their important data in the remote servers in the cloud, without leaving a copy in local computers. Sometimes the data stored in the cloud is so important that the clients must ensure it is not lost or corrupted. While it is easy to check data integrity after completely downloading the data to be checked, downloading large amounts of data just for checking data integrity is a waste of communication bandwidth.

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 order to solve the problem of data integrity checking, many schemes are proposed under different systems and Security models. 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. Remote data integrity checking can be introduced for solving the problems and then propose RSA-based methods for solving this problem. And then have the concept of remote data storage auditing method based on pre-computed challenge-response pairs. Recently many works focus on pro-viding three advanced features for remote data integrity checking scheme. The data dynamics, public verifiability and privacy against verifiers. This scheme can be support data dynamics at the block level, including block insertion, block modification and block deletion and also supports data append operation.

Considering the role of the verifier in the model 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. The outsourced data themselves should not be required by the verifier for the verification.

The following are the problems in the existing strategies: Users’ outsourced data, which inevitably posse’s new security risks towards the correctness of the data in cloud. Dynamic data operations, especially to support block insertion, which is missing in most existing schemes. It does not guarantee the data availability in case of server failures. Cloud Service Provider can monitor your activities. Data Security, Cannot guarantee misuse of data at data centers. Data theft, Hacking is on the increase and all data is exposed on the internet.

More existing in the 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 context of remote data storage mainly focus on static data files and the importance of this dynamic data updates has received limited attention so far. The direct extension of the current provable data possession (PDP) or proof of retrievability (PoR) schemes to support data dynamics may lead to security loopholes.

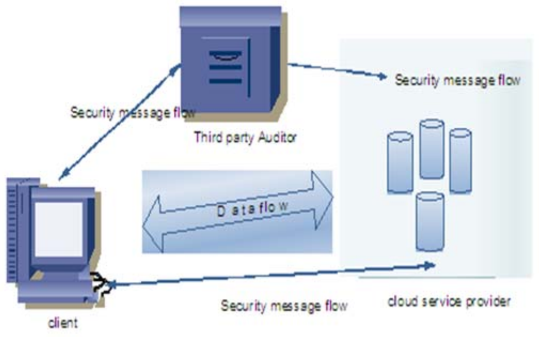

Fig. 1. Cloud data storage architecture.

In this paper, the above problems are addressed by proposing and studying Public audit ability for storage correctness assurance to allow anyone, not just the clients' originally stored the file on cloud servers, to have the capability to verify the correctness of the stored data on demand. 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. Block less verification: no challenged file blocks should be retrieved by the verifier (E.g., TPA)

II. RELATED WORK

The context of remotely stored data verification, 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 holomorphic tags for auditing outsourced data, thus public audit ability is achieved. Do 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. In their subsequent work propose a dynamic version of the prior PDP scheme.

The system imposes a priori bound on the number of queries and does not support fully dynamic data operations, it can only allow very basic block operations with limited functionality, and block insertions cannot be supported. Dynamic datastorage in a distributed scenario, an both determine the data correctness and locate possible errors. And then consider partial support for dynamic data operation. Then the "proof of retrievability" model, where spot-checking and error-correcting codes are used to ensure both "possession" and "retrievability" 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 "sentinels" prevents the development of realizing dynamic data updates. PoR scheme with full proofs of security in the security model. They use publicly verifiable homomorphic authenticators built from BLS signature based on which the proofs can be aggregated into a small authenticator value, and public retriev ability is achieved. They extend the PDP model in to support provable updates tos tored 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 blocki nsertion, they eliminate the index information in the "tag" computation and employ authenticated skip list data structure to authenticate the tagi nformation 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. To achieve a secure and efficient design to seamlessly integrate these two important components for data storage. Generalize the support of data dynamics to both PoR and PDP models and discuss the impact of dynamic data operations.Emphasizethatwhiledynamicdataupdatescanbepe rformedefficientlyin PDP models more efficient protocols need to be designed for the update of the encoded files in PoR models. Data auditing scheme for the single client and explicitly include a concrete description of the multi-client data auditing scheme. And also present the performance comparison between the multi-client data auditing scheme and the individual auditing.

III. THE VERIFICATION SCHEME

A. System Model

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

1) 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.

*2) Cloud Storage Server (CSS):*An entity, which is managed by Cloud Service Provider (CSP), has significant storage space and computation resource o maintain the clients' data.

3) 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 remote servers, the clients can be relieved the burden of storage and computation.

As clients no longer locally, it is of critical importance for the clients to ensure that their data are being correctly stored. 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's necessarily had the time, feasibility or resources to monitor their data, they can delegate the monitoring task to a trusted TPA.

The verification schemes with public audit ability any TPA in possession of the public key can act as a verifier. We 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 prestored data. More importantly, in practical scenarios, the client may perform the block level operations such as modification, insertion, and deletion.

B. Security model

The checking scheme is secure if 1) there exists no polynomial- time algorithm that can cheat the verifier with non-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 interact with the server this scheme is correct if the verification algorithm accepts when interacting with the valid proof and it is sound if any cheating server that convinces the client it is storing the data file is actually storing that file.

Note that in the "game" between the adversary and the client, the adversary has full access to the information stored in the server, i.e., the adversary can play the part of the prover (server). The goal of the adversary is to cheat the verifier successfully, i.e., trying to generate valid responses and pass the data verification without being detected .Our security model has subtle but crucial difference from that of the existing PDP or PoR models in the verification process. As mentioned above, these schemes do not consider dynamic data operations, and the block construction insertion cannot be supported at all.

This is because of the signatures is involved with the file index information i. Therefore, once a file block is inserted, the computation overhead is unacceptable since the signatures of all the following file blocks should be recomputed with the new indexes. To deal with this limitation, we remove the index information i in the computation of signatures.

C. Preserving Public Auditing Scheme

We propose to uniquely integrate the homomorphic authenticator with random masking technique. In our protocol, the linear combination of sampled blocks in the server's response is masked with randomness generated by a pseudo random function (PRF).With random masking, the TPA no longer has all the necessary information to build up a correct group of linear equations and therefore cannot derive the user's data content, no matter how many linear combinations of the same set of file blocks can be collected. Meanwhile, due to the algebraic property of the homomorphic authenticator, the correctness validation of the block-authenticator pairs will not be affected by the randomness generated from a PRF.

D. Construct Public Auditing System

The public auditing system can be constructed in two phases ,Setup and Audit Setup. The user initializes the public and secret parameters of the system by executing KeyGen, and pre-processes the data file F by using SigGen to generate the verification metadata. The user then store the data file F at the cloud server, deletes its local copy, and publishes the verification metadata to TPA for later audit. As part of pre-processing, the user may alter the data file F by expanding it or including additional metadata to be stored at server. The data are received by sign on it Audit.

The TPA issues an audit message or challenge to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message from a function of the stored data file F by executing GenProof. Using the verification metadata, the TPA verifies the response via Verify Proof. Keygen Process have the batch signature scheme based on the BLS signature. The BLS signature scheme uses a cryptographic primitive called pairing, which can be defined as a map over two cyclic groups G1 and G2.

The BLS signature scheme consists of three phases:

1. In the key generation phase, a sender chooses a random integer and computes. The private key is x and the public key is y.
2. Given a message in the signing phase, the sender first computes, where h () is a hash function, and then computes the signature of m.
3. In the verification phase, the receiver first computes and then checks whether. If the verification succeeds, then the message m is authentic. Merits: It can generate a very short signature. It can solve communication overhead.

*1) Problem Formulation :*Denote by m the file that will be stored in the untrusted server, which is divided into n blocks of equal lengths:

$m = m_{1,} m_2 ... m_n$, where $n = |m|/l$.Here l is the length of each file block. Denote by $f_K(\cdot)$ a pseudo-random function which is defined as:

$f : \{0,1\}^k \times \{0,1\}^{\log 2(n)} \rightarrow \{0,1\}^{d,}$in which k and d are two security parameters, and then denote the length of N in bits by |N|. To design a remote data integrity checking protocol that includes the following five functions: Set Up,TagGen, Challenge, GenProof and CheckProof.

SetUp $(1^k) \rightarrow (pk,sk)$: Given the security parameter k, this function generates the public key pk and the secret key sk,pk is public to everyone, while sk is kept secret by theclient.

Tag Gen (pk,sk,m) → Dm: Given pk, sk and m, this function computes a verification tag Dm and makes it publicly known to everyone. This tag will be used for public verification of data integrity.

Challenge $(pk, D_m) \rightarrow$ chal: Using this function, the verifier generates a challenge chal to request for the integrity proof of file m. The verifier sends chal to the server.

Gen Proof (pk, D_m, m, chal) → R: Using this function, the server computes a response R to the challenge. The verifier checks the validity of the response R. If it is valid, the function outputs "success", otherwise the function outputs "failure". The secret key sk is not needed in the Check Proof function.

*2) Merkle hash tree :*A Merkle Hash Tree (MHT) is a well-studied authentication structure which is intended to efficiently and securely prove that a set of elements are undamaged and unaltered. It is constructed as a binary tree where the leaves in the MHT are the hashes of authentic data values. The verifier with the authentic h_r requests for {x_2, x7} and requires the authentication of the received blocks.

The prover provides the verifier with the auxiliary authentication information (AAI) Ω_2<h(x_1), h_d> and Ω_7=<h (xs, he>. The verifier can then verify $x_2$and $x_7$byfirstcomputing h(x2),h(x7),h(h(x1)||h(x2))), hf=h(h(x7||h(x8)),ha=h(hc||hd), hb=h(he||hf) and hr=h(ha||hb),and then checking if the calculated hr is the same as the authentic one. MHT is commonly used to authenticate the values of data blocks. However, in this paper, we further employ MHT to authenticate both the values and the positions of data blocks. We treat the leaf nodes as the left-to-right sequence, so any leaf node can be uniquely determined by following this sequence and the way of computing the root in MHT.

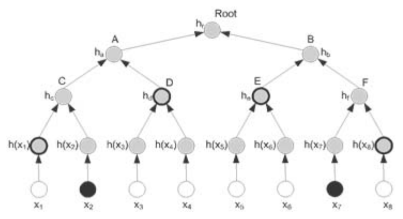

Fig. 2.Merkle Hash Tree

Another basic solution is to use signatures instead of MACs to obtain public auditability.

The data owner Precomputes the signature of each block and sends both F and the signatures to the cloud server for storage.

To verify the correctness of F, the data owner can adopt a spot-checking approach, i.e., requesting a number of randomly selected blocks and their corresponding signatures to be returned. This basic solution can provide probabilistic assurance of the data correctness and support public auditability. However, it also severely suffers from the fact that a considerable number of original data blocks should be retrieved to ensure a reasonable detection probability, which again could result in a large 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.

3) Default Integrity Verification: The client or TPA can verify the integrity of the out sourced data by challenging the server. Before challenging, the TPA first uses pk to verify the signature on t. If the verification fails, reject by emitting FALSE; otherwise, recover u. To generate the message "chal," the TPA (verifier) picks a random c-element subset I = {s_1;s_2;...;s_c} of set [1,n],where we assume s_1<…< s_c. For each i € I the TPA chooses a random element V_i← B© Zp. The message "chal" specifies the positions of the blocks to be checked in this challenge phase. The verifier sends the chal {(i, v_i)}the prover (server).both the data blocks and the corresponding signature blocks are aggregated into a single block, respectively .Inaddition, the prover will also provide the verifier with a small amount of auxiliary information.

*i) Update Operation :*In cloud data storage, sometimes the user may need to modify some data block(s) stored in the cloud, from its current value fij to a new one, fij + Δ fij. We refer this operation as data update.

ii) Delete Operation: Sometimes, after being stored in the cloud, certain data blocks may need to be deleted. The delete operation we are considering is a general one, in which user replaces the data block with zero or some special reserved data symbol. From this point of view, the delete operation is actually a special case of the data update operation, where the original data blocks can be replaced with zeros or some predetermined special blocks.

iii) Append Operation: In some cases, the user may want to increase the size of his stored data by adding blocks at the end of the data file, which we refer as data append. We anticipate that the most frequent append operation in cloud data storage is bulk append, in which the user needs to upload a large number of blocks at one time.

4) Designs for distributed data storage security: To further enhance the availability of the data storage security,individual user's data can be redundantly stored in multiple physical locations. That is, besides being exploited individual servers, data redundancy can also be employed across multiple servers to tolerate faults or server crashes a suser's data grow in size and importance. It is well known that erasure-correcting code can be used to tolerate multiple failures in distributed storage systems.

In cloud datastorage, we can rely on this technique to disperse the data file F redundantly across a set of n ¼ m þ k distributed servers. A ½m þ k;k?-Reed-Solomon code is used to create k redundancy parity vectors from m data vectors in such away that the original m data vectors can be reconstructed from any m out of the m þ k data and parity vectors. By placing each of the m þ k vectors on a different server, the original data file can survive the failure of any k of them þ k servers without any data loss. Such a distributed cryptographic system allows a set of servers to prove to a client that a stored file is intact and retrievable.

IV. Performance Analysis

Make a comparison of our scheme and the state of the art.The scheme in [14] extends the original PDP [2] to support data dynamics using authenticated skip list. Thus, we call it DPDP scheme thereafter. For the sake of completeness, we implemented our BLS and RSA-based instantiations as well as the state-of-the-art scheme in Linux.

Our experiment is conducted using C on a system with an Intel Core 2 processor running at 2.4 GHz, 768 MB RAM, and a7200 RPM Western Digital 250 GB Comparisons of Different Remote Data Integrity Checking Schemes. The security parameter is eliminated in the costs estimation for simplicity. The scheme only supports bounded number of integrity challenges and partially data updates, i.e., data insertion is not supported. y No explicit implementation of public auditability is given for this scheme.8 MB buffer are implemented using the Pairing-Based Cryptography (PBC)library version 0.4.18 and the crypto library of Open SSLversion 0.9.8h. To achieve 80-bit security parameter, the curve group we work on has a 160-bit group order and the size of modulus N is 1,024 bits. All results are the averages of 10 trials.

The performance metrics for 1 GBfile under various erasure code Due to the smaller block size compared to RSA-based instantiation, our BLS-based instantiation is more than two times faster than the other two in terms of server computation time larger computation cost at the verifier side as the pairing operation in BLS can perform scheme consumes more time than RSA techniques. The communication cost of DPDP scheme is the largest among the three in practice. The tuple values associated with each skip list node for one proof, which results in extra communication cost as compared to our constructions. The communication over-head of our RSA-based instantiation and DPDP scheme under different block. We can see that the communication cost grows almost linearly as the block size increases, which is in caused by the increasing in size of the verification block. However, the experiments suggest that when block size is chosen around16 KB, both schemes can achieve an optimal point that minimizes the total communication cost.

Conduct experiments for multi-client batch auditing and demonstrate its efficiency, where the number of clients in the system is increased from 1 to 100 with intervals of 4. A batch auditing not only enables simultaneously verification from multiple-client, but also reduces the computation cost on the TPA side. Given total K clients in the system, the batch auditing equation helps reduce the number of expensive pairing operations from 2K, as required in the individual auditing, to K þ 1.

Thus, a certain amount of auditing time is expected to be saved. Specifically, following the same experiment setting a ρ=99% and 97%, batch auditing indeed saves TPA's computation overhead for about 5 and14 % respectively. Note that in order to maintain detection probability of 99% the random sample size in TPA's challenge for ρ=99% is quite larger than ρ= 97,as this sample size is also a dominant factor of auditing time, this explains why batch auditing for ρ= 99% is not as efficient as for ρ= 97%.

V. Conclusion

The public auditability for cloud data storage security is of critical importance so that users can resort to an external audit party to check the integrity of outsourced data when needed. This work studies the problem of ensuring the integrity of data storage in Cloud Computing. In particular, we consider the task of allowing a third party auditor (TPA), on behalf of the cloud client, to verify the integrity of the dynamic data stored in the cloud. We utilize and uniquely combine the public key based homomorphic authenticator with random masking to achieve the privacy-preserving public cloud data auditing system.

This scheme is the first to support scalable and efficient public auditing in the Cloud Computing. The technique of Bilinear Aggregate signature is used to achieve batch auditing, where TPA can perform multiple auditing tasks simultaneously. The data in the cloud does not remain static. Unlike most prior works, the new scheme further supports secure and efficient dynamic operations on data blocks stored in the cloud, including: data update, delete and append.

References

[1]. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, (2007), 'Provable Data Possession at Untrusted Stores', Proc. IEEE /ACM pp. 598-609.

[2]. Bowers, A. Juels, and A. Oprea (2009),'Hail: A High-Availability and Integrity Layer for Cloud Storage', Proc. IEEE in computer communication pp. 187-198.

[3]. C. Chang and J. Xu, (2008), 'Remote Integrity Check with Dishonest Storage Server', Proc IEEE/ACM, pp. 223-237.

[4]. Craig Gentry, Dan Boneh, (2004) ,"Aggregate and verifiably encrypted signatures from bilinear maps".

[5]. Craig Gentry, Dan Boneh, (2004) ,"Aggregate and verifiably encrypted signatures from bilinear maps".

[6]. Ferrara, M. Greeny, S. Hohenberger, M. Pedersen (2009), "Practical short signature batch verification".

[7]. Juels.A and B.S. Kaliski Jr., (2007), 'Pors: Proofs of Retrievability for Large Files', IEEE in computer and communication security, pp. 584-597.

[8]. M.A. Shah, R. Swaminathan, and M. Baker (2008), 'Privacy-Preserving Audit and Extraction of Digital Contents', IEEE Cryptology Archive.

[9]. T. Schwarz and E.L. Miller,(2006) "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage".

[10]. Wang, K. Ren, W. Lou (2010), "Achieving secure, scalable, and fine-grained access control in cloud computing

[11]. ".