International Journal of Emerging Technologies in Engineering Research (IJETER)
Volume 5, Issue 12, December (2017) www.ijeter.everscience.org

Secure and Efficient Third Party Auditing Scheme for Cloud Data Storage

Manibharathi R
Senior Developer, Shiro Software Solutions, Nagercoil, Tamil Nadu, India.

Dr.K.Selva kumar
Assistant Professor, Department of Mathematics, University College of Engineering, Nagercoil, Tamil Nadu, India.

Dinesh R
Project Manager, Shiro Software Solutions, Nagercoil, Tamil Nadu, India.

Abstract – Cloud computing is the long dreamed vision of computing as a utility, where users can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources. However, the fact that users no longer have physical possession of the possibly large size of outsourced data makes the data integrity protection in Cloud computing a very challenging and potentially formidable task, especially for users with constrained computing resources and capabilities. Existing privacy-preserving public auditing protocols assume the end devices of users are powerful enough to compute all costly operations in real time when the data to be outsourced is given. In fact, the end devices may also be those with low computation capabilities. Thus the proposed system is used for two lightweight privacy-preserving public auditing protocols. 1) TPA should be able to efficiently audit the cloud data storage without demanding the local copy of data, and introduce no additional on-line burden to the cloud user; 2) The third party auditing process should bring in no new vulnerabilities towards user data privacy. The algorithm proposed here is online/offline algorithm. Thus the proposals support batch auditing and data dynamics.

Index Terms – Cloud Computing, Big data, Secure Storage, Significance,

1. INTRODUCTION

Cloud computing is a new-generation distributed computing platform that is extremely useful for data storage and processing. Many data applications are being migrated or have been migrated into clouds. One of thecloud's core concepts is 'X as a Service' (XaaS), including Infrastructure-as-a-Service(IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS), which means that both individual and enterprise users can use IT services in a pay-as-you-go fashion. Compared to traditional distributed systems, cloud computing brings outstanding advantages Times New Roman font with size 10 should be used.

Many international IT partnerships offer these administrations to clients on a scale from individual to big business everywhere throughout the world; cases are Amazon AWS, Microsoft Azure, and IBM Smart Cloud.

Information reviewing plans can empower cloud clients to check the respectability of their remotely put away information without downloading them locally, which is named as blockless confirmation. With inspecting plans, clients can occasionally collaborate with the CSP through reviewing conventions to check the accuracy of their outsourced information by confirming the respectability confirmation figured by the CSP, which offers more grounded trust in information security since client's own decision that information is in place is substantially more persuading than that from specialist organizations. For the most part talking, there are a few patterns in the improvement of evaluating plans.

Above all else, prior evaluating plans more often than not require the CSP to create a deterministic verification by getting to the entire information record to perform honesty check, e.g., plots in [1], [2] utilize the whole document to perform particular exponentiations. Such plain arrangements bring about costly calculation overhead at the server side, thus they need productivity and common sense when managing vast size information. Spoken to by the "testing" technique in "Confirmations of Retrievability" (PoR) [3] display and "Provable Data Possession" (PDP) [4] demonstrate, later plans [5], [6] have a tendency to give a probabilistic evidence by getting to some portion of the record, which clearly improves the examining proficiency over prior plans.

Data security and information isolation are a portion of the fundamental worry in the acknowledgment of cloud computing. Clients lose their immediate control on information when they aggregate information on cloud as contrast with traditional frameworks. The issue of genuineness affirmation for information stockpiling on cloud called as information review when the affirmation is directed by a trusted outsider i.e. TPA called as an assessor.

Furthermore, some examining plans [3], [7] give private obviousness that require just the information proprietor who has the private key to play out the inspecting errand, which may possibly overburden the proprietor because of its restricted calculation capacity. Ateniese el al. [4] were the first to propose to empower open unquestionable status in reviewing plans. Conversely, open inspecting plans [5], [6] permit any individual who has the general population key to play out the examining, which makes it feasible for the evaluating undertaking to be appointed to an outside outsider examiner (TPA). A TPA can play out the honesty mind benefit of the information proprietor and genuinely report the inspecting result to him [8].

Generally, this paper proposes a new auditing scheme to address the problems of data dynamics support, public verifiability and dispute arbitration simultaneously. Our contributions mainly lie in:

2. RELATED WORK

Remote integrity check could be sourced to memory check schemes [21], [22] that aim to verify read and write operations to a remote memory. Recently, many auditing schemes [1], [2], [23], [24], [25], [26] have been proposed around checking the integrity of outsourced data.

Deswarte et al. [1] and Filho et al. [2] use RSA-based hash functions to check a file's integrity. Although their approaches allow unlimited auditing times and offer constant communication complexity, their computation overhead is too expensive because their schemes have to treat the whole file as an exponent. Opera et al. [23] propose a scheme based on tweakable block cipher to detect unauthorized modification of data blocks, but verification needs to retrieve the entire file, thus the overhead of data file access and communication are linear with the file size. Schwarz et al. [24] propose an algebraic signature based scheme, which has the property that the signature of the parity block equals to the parity of the signatures on the data blocks. However, the security of their scheme is not proved. Sebe et al. [26] provide an integrity checking scheme based on the Diffie- Hellman problem. They fragment the data file into blocks of the same size and fingerprint each data block with an RSAbased hash function. But the scheme only works when the block size is much larger than the RSA modulus N, and it still needs to access the whole data file. Shah et al. [7], [27] propose a privacy-preserving auditing protocol that allows a third party auditor to verify the integrity of remotely stored data and assist to extract the original data to the user. As their scheme need firstly encrypt the data and precompute a number of hashes, the number of auditing times is limited and it only works on encrypted data. Furthermore, when these hash values are used up, the auditor has to regenerate a list of new hash values, which leads to extremely high communication overhead.

From above analysis, it can be seen that earlier schemes usually generate a deterministic proof by accessing the whole data file, thus their efficiency is limited due to the high computation overhead. To address this problem, later schemes tend to generate a probabilistic proof by accessing part of the date file. Jules et al. [3], [28] propose a proofs of retrievability (PoR) model, where spot-checking and errorcorrecting code are used to guarantee the possession and retrievability of remote stored data. However, PoR can only be applied to encrypted data, and the number of auditing times is a fixed priori due to the fact that sentinels embedded in the encrypted data could not be reused once revealed. Dodis el al. identify several other variants of PoR in [29]. Ateniese et al. [4] are the first to put forward the notion of public verifiability in their provable data possession (PDP) scheme, where the auditing tasks can be delegated to a third-party auditor. In PDP, they propose to randomly sample a few data blocks to obtain a probabilistic proof, which greatly reduces the computation overhead. Moreover, PDP scheme allows unlimited number of auditing. Shacham et al. [5] design an improved PoR scheme and provide strict security proofs in the security model defined in [3], they use homomorphic authenticators and provable secure BLS signatures [30] to achieve public verifiability, which is not provided in Jules' main PoR scheme. Some other schemes [7], [14], [31] with public auditability aim to provide privacy protection against information leakage toward a third-party auditor in the process of integrity auditing.

To support data dynamics in auditing schemes, Ateniese et al. [32] propose a dynamic version of their original PDP scheme using symmetric encryption, however, the number of auditing times is limited and fully block insertion is not supported (only append-type insertion is supported). Erway et al. [9] firstly propose to construct a fully dynamic provable data possession (DPDP) scheme. To eliminate the index limitation of tag computation in original PDP scheme and avoid tag re-computation brought by data dynamics, they use the rank of a skip list node (similar to block index) to uniquely differentiate among blocks and authenticate the tag information of challenged blocks before proof verification. However, the skip list in essence is an authenticated structure used to test set-membership for a set of elements. To prove the membership of a specific node, a verification path from the start node to the queried node must be provided, its communication cost is linear to the number of challenged blocks. Moreover, there's no explicit implementation of public verifiability given for their scheme. Qian Wang et al. [6] combine BLS signature based homomorphic authenticator with Merkle hash tree to provide both public auditability and fully dynamic operations support. Specifically, their scheme constructs a Merkle hash tree, stores the hashes of tags in the leaf nodes and recursively computes the root and signs it, which is used to authenticate the tags of challenged blocks.

Furthermore, they eliminate the index limitation in tag computation by using H(mi) to replace H(name||i) in [5], which requires blocks to be different with each other. However, such a requirement on data blocks is not appropriate since the probability of block resemblance increases when block size decreases. In addition, due to the authentication of challenged blocks with a Merkle Hash Tree [33], the communication cost of their scheme is also linear to the number of requested blocks. Zhu et al. [10], [34] use index-hash table to construct their dynamic auditing scheme based on zero knowledge proof, which is similar to our index switcher in terms of index differentiation and avoidance of tag re-computation. But their design mainly focuses on data dynamics support, while our scheme goes further by achieving dynamic operations support and fair arbitration together.

Qian Wang et al. [6] combine BLS signature based homomorphic authenticator with Merkle hash tree to provide both public auditability and fully dynamic operations support. Specifically, their scheme constructs a Merkle hash tree, stores the hashes of tags in the leaf nodes and recursively computes the root and signs it, which is used to authenticate the tags of challenged blocks. Furthermore, they eliminate the index limitation in tag computation by using H(mi) to replace H(name||i) in [5], which requires blocks to be different with each other. However, such a requirement on data blocks is not appropriate since the probability of block resemblance increases when block size decreases. In addition, due to the authentication of challenged blocks with a Merkle Hash Tree [33], the communication cost of their scheme is also linear to the number of requested blocks. Zhu et al. [10], [34] use index-hash table to construct their dynamic auditing scheme based on zero-knowledge proof, which is similar to our index switcher in terms of index differentiation and avoidance of tag re-computation. But their design mainly focuses on data dynamics support, while our scheme goes further by achieving dynamic operations support and fair arbitration together.

Recently, providing fairness and arbitration in auditing schemes has become an important trend, which extends and improves the threat model in early schemes to achieve a higher level of security insurance. Zheng et al. [11] construct a fair and dynamic auditing scheme to prevent a dishonest client accusing an honest CSP. But their scheme only realizes private auditing, and is difficult to be extended to support public auditing. Kupcu [12] proposes a framework on top of Erway's DPDP scheme [9], where the author designs arbitration protocols on the basis of fair signature exchange protocols in [13]. Moreover, the author goes further by designing arbitration protocols with automated payments through the use of electronic cash. Compared to these schemes, our work is the first to combine public verifiability, data dynamics support and dispute arbitration simultaneously.

Other extensions to both PDPs and PoRs are given in [35], [36], [37], [38], [39]. Chen et al. [37] introduce a mechanism for data integrity auditing under the multiserver scenario, where data are encoded with network code. Curtmola et al. [35] propose to ensure data possession of multiple replicas across the distributed storage scenario. They also integrate forward error-correcting codes into PDP to provide robust data possession in [36]. Wang et al. [39] utilize the idea of proxy re-signatures to provide efficient user revocations, where the shared data are signed by a group of users. And in [16], [38], they exploit ring signatures to protect the identity-privacy of signers from being known by public verifiers during the auditing.

3. PROPOSED MODELLING

In the cloud environment, both clients and CSPs have the motive to cheat. The scheme in the aim of supporting variable-sized data blocks, authorized third party auditing and fine-grained dynamic data updates is described in four parts which are shown in figure 4.1ist as follows:

User: Users are those who have data to be stored in the cloud.

Cloud Service Provider (CSP): A CSP, who has significant resources and expertise in building and managing distributed cloud storage servers.

Third Party Auditor (TPA): A TPA is trusted to assess and expose risk of cloud storage services based on the user's request.

Third Party Arbitrator (TPAR): A TPAR is trusted to fairly settle any dispute about proof verification and dynamic update, and find out the cheating party.

To guarantee secure access control in public cloud storage, we claim that an access control scheme needs to meet the following four basic security requirements:

Data confidentiality. Data content must be kept confidential to unauthorized users as well as the curious cloud server.

Collusion-resistance. Malicious users colluding with each other would not be able to combine their attributes to decrypt a ciphertext which each of them cannot decrypt alone.

TPA accountability. An auditing mechanism must be devised to ensure that an TPAs misbehavior can be detected to prevent Users abusing their power without being detected.

No ultra vires for any TPA. An TPA should not have unauthorized power to directly generate secret keys for users. This security requirement is newly introduced based on our proposed hierarchical framework.

- The owner of the file will upload the data to the cloud[The file is fragmented into blocks, encrypted and stored in cloud].Each file is recognized by a unique ID and the details about file ID and its associated block data are maintained in a log file in cloud by CSS[for each file the log detail is updated].

- After uploading, the owner of the file also plays a role of client as any authenticated client can update /modify the existing file in cloud.

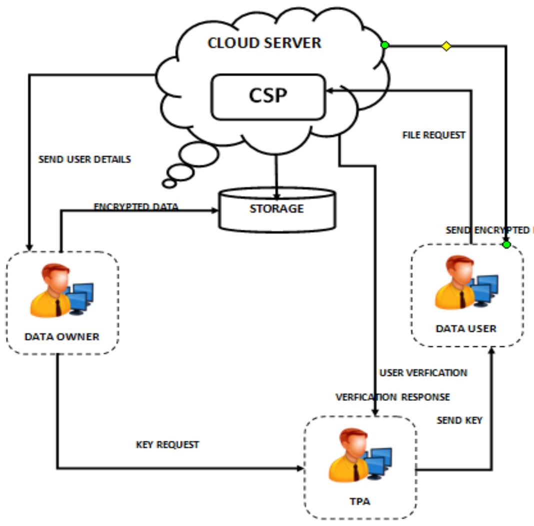

Fig 1. Architecture

- When the file gets uploaded to the cloud the third party auditor (TPA) verifies the file and its blocks [e.g File consists of all pages from 1 to 10 or not].If the file is consistent, then verified flag is set with unique key value.
- Similarly the CSS also verifies the file, the same way as (TPA) and the generated key will also be similar, if the file is consistent.
- The client after logging in to the cloud can get to see about the files available, but the client cannot access it.
- Before giving access to the client the third party arbitrator authenticates the file by matching the key provided by TPA and CSS to TPAR; if it matches, then the client is given access to the file.
- Now here the client also has the privilege to modify the file.This is the reason why the owner is also logged in as a client.
- After the updation of the file,step 3 to step 7 are followed the same.
- For each client's request,before giving access to the client,the TPAR authenticates and then proceeds accordingly.

2.1 Design Goals

Our design goals can be summarized as follows:

1) Public verifiability for data storage correctness: to allow anyone who has the public key to verify the correctness of users' remotely stored data;

2) Dynamic operation support: to allow cloud users to perform full block-level operations (modification, insertion and deletion) on their outsourced data while guarantee the same level of data correctness, and the scheme should be as efficient as possible;

3) Fair dispute arbitration: to allow a third party arbitrator to fairly settle any dispute about proof verification and dynamic update, and find out the cheating party.

4. PROPOSED METHODOLOGY

Setup: The client will generate keying materials via KeyGen and FileProc, and then upload the data to CSS. The client will store a RMHT instead of a MHT as metadata. Moreover, the client will authorize the TPA by sharing a value signature with AUTHENTICATON.

Verifiable Data Updating: The CSS performs the client's fine-grained update requests via Perform Update,then the client runs VerifyUpdate tocheck whether CSS has performed the updates on both the data blocks and their corresponding authenticators (used for auditing) honestly.

Challenge, Proof Generation and Verification:It describes how the integrityof the data stored on CSS is verified by TPA via GenChallenge,GenProof and Verify.

3.1 Integrity Checking

TPA must show CSS that it is indeed authorized by the file owner before it can challenge a certain file. TPA will decide to verify some blocks from the total blocks. Then, a challenge message is generated with TPA's ID, which is encrypted with the CSS's public key. So that CSS can later decrypt with the corresponding secret key. TPA then sends challenge to CSS. After receiving challenge, CSS will first verify signature and the client's public key and output REJECT if it fails. Otherwise, CSS will compose the proof P as then output P. CSS will send P to TPA. After receiving P, TPA verify signature by using public keys. If they are equal, then returns TRUE, otherwise it returns FALSE.

3.2 Block-Level Operations

- Block-level operations in fine-grained dynamic data updates may contain the following 5 types of operations
- Partial Modification: Consecutive part of a certain block needs to be updated;

- Whole-Block Modification: Whole block needs to be replaced by a new set of data;
- Block Deletion: Whole block needs to be deleted from the tree structure;
- Block Insertion : Whole block needs to be created on the tree structure to contain newly inserted data;
- Block splitting: Part of data in a block needs to be taken out to form a new block to be inserted next to it.

3.3 Arbitration

As we have called attention to previously, in the cloud condition, the two customers and CSPs have the thought process to swindle. In our plan, the PPAP is utilized by the reviewer to get key lists for asked for obstructs at evidence confirmation stage, hence the check result depends on the accuracy of the record maker. Be that as it may, the age and refresh of record switcher are performed by the information proprietor just, it will possibly give an untrustworthy proprietor the chance of dishonestly blaming a genuine CSP. In this sense, we should give some system to guarantee the accuracy of the file switcher and further the reasonableness of conceivable assertion, so no gathering can outline the other party without being recognized.

A direct route is to let the arbitrator(TPAR) keep a duplicate of the list of keys. Since the difference in the list is caused by unique tasks, the customer can send essential refresh data (i.e., activity write, activity position, new label file) to the TPAR for each refresh activity. With these data, the referee could re-develop the most recent variant of the file switcher, whose accuracy chooses the legitimacy of later assertion. In any case, such an answer costs O(n) stockpiling at the authority side and needs the mediator to be associated with each refresh activity. In a perfect world, we need the TPAR just embrace the part of an authority who includes just at question settlement, and keeps up a consistent stockpiling for state data, i.e., open keys of the customer and the CSP.

3.3.1 Integrity Proof

1) The Third party Arbitrator (TPAR) requests {$Seq_{c;}\Omega_s$; Sig_s} and Sig_s from the client. Then he checks the signature Sig_s of the CSP. If it is invalid, the TPAR may unauthorize the client for their ineffiency; otherwise the TPAR proceeds.

2) The TPAR requests {Seq_s;Ω_s; Sig_c} from the CSP. Then he checks the signature Sig_c of the client. If the signature does not verify correctly, the TPAR may unauthorize the CSP for their ineffiency; otherwise the TPAR proceeds.

3) If $Seq_c = Seq_s$, then the TPAR requests from the client the challenged set Q that causes dispute on proof verification and retransmit it to the CSP to run the auditing scheme. The CSP computes the proof returns it to the TPAR for verification. The TPAR checks the proof using the verified index switcher.

4) If there is a mismatch in Seq_c and Seq_s. The TPAR can be sure that the party who gives a smaller sequence number is performing a replay attack; he may unauthorize the cheating party. Specifically, if Seq_c>Seq_s, the client is cheating by replaying an old signature from the CSP; if Seq_s>Seq_c, the CSP is cheating by replaying an old signature from the client.

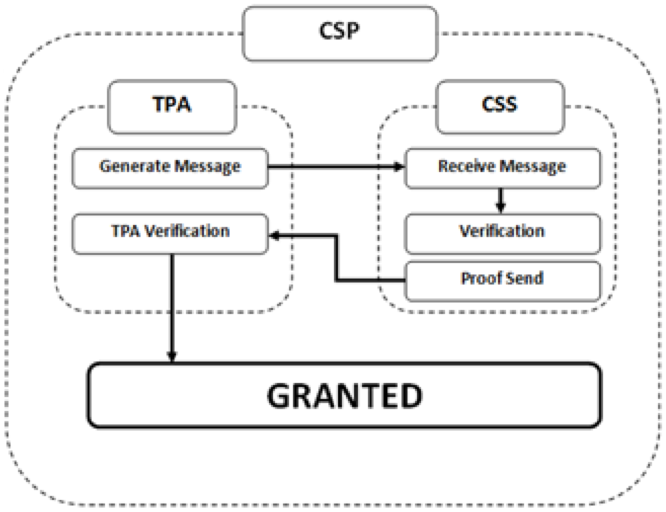

Fig 2. Message Generated System

3.3.2 Fair Arbitration Dynamic Update

The first two steps are the same as that of the arbitration protocol on integrity proof. According to the result of sequence number comparison (Seqc and Seqs), we divide the protocol into two situations.

- The TPAR requests the update record from the client.
- For block modification and insertion, the TPAR verifies the correctness of by verifying. If fails, the TPAR may unauthorized the client for cheating; otherwise, the TPAR is convinced that the updated block and its tag are consistent with each other. For block deletion, this step can be omitted.
- The TPAR transmits to the CSP, and requests on the small challenge set from the CSP. Then he verifies the validity according to algorithm. If fails, the TPAR may unauthorize the CSP for denying the update; otherwise, the TPAR proceeds.
- The TPAR updates the index switcher, and then he requests and verifies new signatures from both parties. The TPAR may unauthorize the party who

sends an invalid signature. If both signatures verify, the TPAR forwards to the CSP, and to the client.

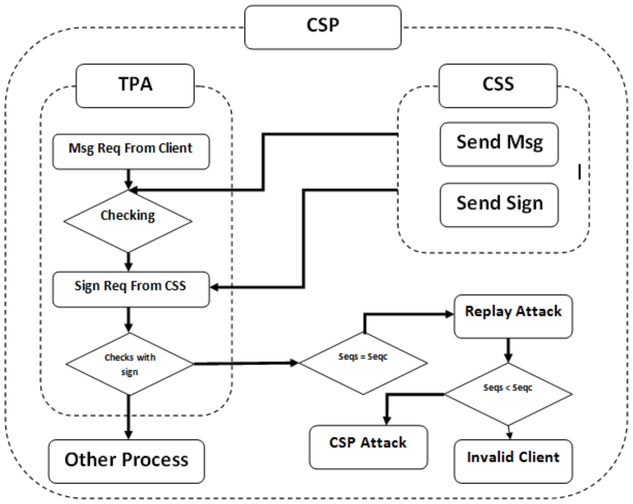

Fig 3. Message Checking System

5. PERFORMANCE EVALUATION

Our scheme is implemented using Scripting language on a Linux system equipped with a 4-Core Intel 3 processor running at 2.4GHz, 4GB RAM and a 7200 RPM 2TB drive. Algorithms are implemented using the Pairing-Based Cryptographic (PBC) library 0.5.11 and the crypto library OpenSSL 1.0.0. For security parameters, we choose the curve group with a 160-bit group order and the size of modulus is 1024 bits. Our scheme provides probabilistic proof as [4]: if t fraction of the file is corrupted, by challenging a constant c blocks of the file, the auditor can detect the data corruption behavior at least with probability p = 1 − (1 − t)c. We choose c = 460, thus the detection probability is about 99%. The integrity proof is of constant size as in [5]. While most authenticated structure based schemes [6], [9], [11] need to send auxiliary authentication information to the auditor, which leads to linear communication overhead. The size of the test data is 10 GB, and the block size of fragmentation varies from 2 KB to 1 MB. All results are on the average of 10 trials.

We analyze the performance of our auditing scheme from three aspects: key generation time, encryption time time and File Upload time. For data dynamic update and dispute arbitration, we test the update overhead by inserting, deleting and modifying 100 blocks and tags. In addition, we test the cost of signature computation and verification with the index switcher containing different number of index pairs, and the shifting overhead of index pairs caused by block insertion and deletion.

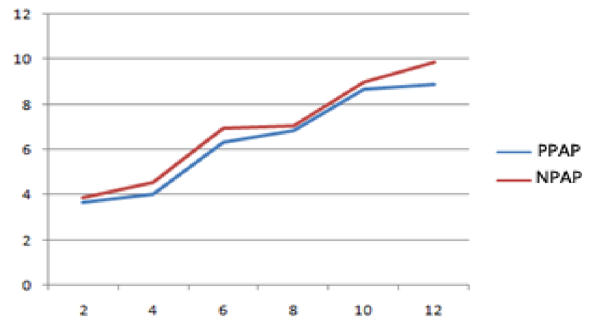

Fig 4. File Size vs Upload Time

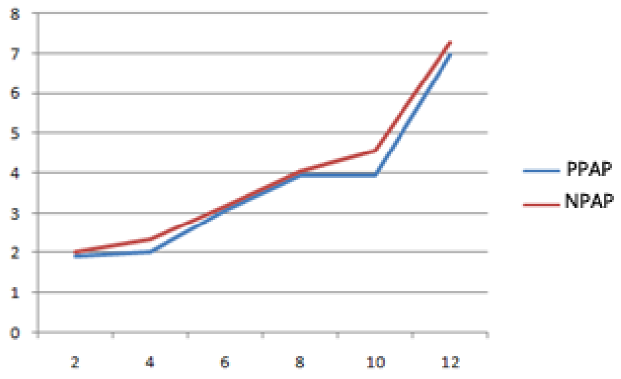

Fig 5. File Size Vs Encryption Time

6. CONCLUSION

The aim of this paper is to provide an honesty evaluating plan with public undeniable nature, proficient information flow and reasonable debate intervention. To dispense with the confinement of list use in label calculation and proficiently support information flow, we separate between piece records and label files, and devise a list switcher to keep square label list mapping to stay away from label re-calculation caused by square refresh activities, which brings about constrained extra overhead, as appeared in our execution assessment. In the mean time, since the two customers and the CSP possibly may get rowdy amid inspecting and information refresh, we expand the current danger show in flow research to give reasonable intervention to understanding question amongst customers and the CSP, which is of essential criticalness for the sending and advancement of evaluating plans in the cloud environment. Our examinations show the effectiveness of our proposed scheme, whose overhead for dynamic update and dispute arbitration are reasonable.

REFERENCES

[1] Y. Deswarte, J.-J. Quisquater, and A. Sa¨ıdane, "Remote integrity checking," in Proc. 5th Working Conf. Integrity and Intl Control in Information Systems, 2004, pp. 1–11.
[2] D. L. Gazzoni Filho and P. S. L. M. Barreto, "Demonstrating data possession and uncheatable data transfer." IACR Cryptology ePrint Archive, Report 2006/150, 2006.
[3] A. Juels and B. S. Kaliski Jr, "Pors: Proofs of retrievability for large files," in Proc. 14th ACM Conf. Computer and Comm. Security (CCS07), 2007, pp. 584–597.
[4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Computer and Comm. Security (CCS07), 2007, pp. 598–609.
[5] H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. 14th Intl Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT 08), 2008, pp. 90–107.
[6] 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. 14th European Conf. Research in Computer Security (ESORICS 08), 2009, pp. 355–370.
[7] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents." IACR Cryptology ePrint Archive, Report 2008/186, 2008.
[8] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," Network, IEEE, vol. 24, no. 4, pp. 19–24, 2010.
[9] C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. 16th ACM Conf. Computer and Comm. Security (CCS 09), 2009, pp. 213–222.
[10] Y. Zhu, H.Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in Proc. ACM Symp. Applied Computing (SAC 11), 2011, pp. 1550–1557.
[11] Q. Zheng and S. Xu, "Fair and dynamic proofs of retrievability," in Proc. 1st ACM Conf. Data and Application Security and Privacy (CODASPY 11), 2011, pp. 237–248.
[12] A. K¨upc¸ ¨u, "Official arbitration with secure cloud storage application," The Computer Journal, pp. 138–169, 2013.
[13] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," in Proc. 17th Intl Conf. Theory and Applications of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT98), 1998, pp. 591–606.
[14] C.Wang, Q.Wang, K. Ren, andW. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in Proc. IEEE INFOCOM, 2010, pp. 1–9.
[15] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacypreserving public auditing for secure cloud storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362–375, 2013.
[16] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," IEEE Trans. Cloud Computing, vol. 2, no. 1, pp. 43–56, 2014.
[17] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in Proc. 22nd Intl Conf. Theory and Applications of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT03), 2003, pp. 416–432.
[18] P. A. Bernstein and N. Goodman, "An algorithm for concurrency control and recovery in replicated distributed databases," ACM Trans. Database Systems, vol. 9, no. 4, pp. 596–615, 1984.
[19] J. Hendricks, G. R. Ganger, and M. K. Reiter, "Low-overhead by zantine fault-tolerant storage," in ACM SIGOPS Operating Systems Review, vol. 41, no. 6, 2007, pp. 73–86.
[20] J. Gray, P. Helland, P. O'Neil, and D. Shasha, "The dangers of replication and a solution," in ACM SIGMOD Record, vol. 25, no. 2, 1996, pp. 173–182.
[21] M. Blum, W. Evans, P. Gemmell, S. Kannan, and M. Naor, "Checking the correctness of memories," Algorithmica, vol. 12, no. 2-3, pp. 225–244, 1994.
[22] M. Naor and G. N. Rothblum, "The complexity of online memory checking," in Proc. 46th Ann. IEEE Symp. Foundations of Computer Science, 2005, pp. 573–582.
[23] A. Oprea, M. K. Reiter, and K. Yang, "Space-efficient block storage integrity." in Proc. 9th Network and Distributed System Security Symp. (NDSS '05), 2005.
[24] T. S. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in Proc. IEEE Intl Conf. Distributed Computing Systems (ICDCS 06), 2006, pp. 12–12.
[25] E.-C. Chang and J. Xu, "Remote integrity check with dishonest storage server," in Proc. 13th European Conf. Research in Computer Security (ESORICS 08), 2008, pp. 223–237.
[26] F. Seb´e, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 8, pp. 1034–1038, 2008.
[27] M. A. Shah, M. Baker, J. C. Mogul, R. Swaminathan et al., "Auditing to keep online storage services honest." in Proc. 11th USENIX Workshop Hot Topics in Operating Systems (HotOS 07), 2007, pp. 1–6.
[28] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: Theory and implementation," in Proc. ACM Cloud Computing Security Workshop (CCSW 09), 2009, pp. 43–54.
[29] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in Proc. Theory of cryptography(TCC '09), 2009, pp. 109–127.
[30] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in Proc. 7th Intl Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT 01), 2001, pp. 514–532.
[31] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," IEEE Trans. Knowledge and Data Eng., vol. 23, no. 9, pp. 1432–1437, 2011.
[32] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. 4th Intl Conf. Security and Privacy in Comm. Networks (SecureComm 08), 2008, pp. 1–10.
[33] R. C. Merkle, "Protocols for public key cryptosystems," in Proc. IEEE Symp. Security and Privacy, 1980, pp. 122–133.
[34] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231–2244, 2012.
[35] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "Mr-pdp: Multiple-replica provable data possession," in Proc. 28th Int'l Conf. Distributed Computing Systems (ICDCS '08), 2008, pp. 411–420.
[36] R. Curtmola, O. Khan, and R. Burns, "Robust remote data checking," in Proc. 4th ACM int'l Workshop on Storage Security and Survivability, 2008, pp. 63–68.
[37] B. Chen, R. Curtmola, G. Ateniese, and R. Burns, "Remote data checking for network coding-based distributed storage systems," in Proc. ACM Cloud Computing Security Workshop (CCSW 10), 2010, pp. 31–42.
[38] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," in Proc. 5th Int'l Conf. Cloud Computing, 2012, pp. 295–302.
[39] "Panda: Public auditing for shared data with efficient user revocation in the cloud," IEEE Trans. Services Computing, vol. 8, no. 1, pp. 92–106, 2013.

Authors

Mr. R. Manibharathi completed MCA from Institute Of Road and Transport Technology, Erode, Anna University, Chennai. He graduated BSc Information Technlogy from Manonmaniam Sundaranar University. He is currently working as Senior Developer at Shiro Software Solutions. His research interest in Cloud Computing, networking and Big Data.

Dr. K. Selva kumar started my research and teaching works from April 1987 at Bharathidasan University, Trichy, Tamilnadu, India. Received Ph.D. from Bharathidasan University in 1992. At present working as Assistant Professor in Mathematics at University College of Engineering, Anna University, Nagercoil Campus, Tamilnadu, India. Developing software for Networking , cloud computing. Image Processing, Singular perturbation problems in control design, aircraft optimal control guidance, Numerical methods for engineering related research problems.

Mr. Dinesh R has obtained his Bachelor Degree in Physics from Manonmaniam Sundaranar University. The obtained his Masters Degree from Anna University. Currently He is working as Project Manager in Shiro Software Solutions. He has 6 years experience in Software industries. He also has 4 years experience in teaching as Assistant Professor. His Specializations include Cloud Computing, Big data and Networkig.