Research Paper Engineering

Privacy Preserving and Public Auditing Service for Data Storage in Cloud Computing

*V. Tejaswini **K. Sunitha ***S.K. Prashanth

*,** M.Tech (C.S.E) VCE (AICTE), Hyderabad, India

** Prof. CSE, VCE (AICTE), Hyderabad India

ABSTRACT

The cloud computing model represents a new paradigm shift in internet-based services that delivers highly scalable distributed computing platforms in which computational resources are offered 'as a service'. Security is considered one of the top ranked open issues in adopting the cloud computing model includes data Integrity confidentiality. Wang proposed a enabling public audit ability and data dynamics for storage security in cloud computing. They achieved the integrity guarantee of data storage with support of public audit ability and dynamic data operations. However their protocol lacks in providing privacy of data which is one of the issue for the cloud data storage. In this we proposed a privacy preserving public verifiability for integrity of data storage in cloud computing. We are using RSA public cryptography to provide confidentiality of data. Our scheme is more secure than existing system.

Keywords: Data Integrity, Data Confidentiality, Cloud Computing, RSA

1. Introduction

Cloud model represents a new paradigm shift in internet-based services that delivers highly scalable distributed computing platforms in which computational resources are provisioned 'as a service'. This new data storage "Cloud" brings about many challenging issues which have deep influence on the security and performance of the total system. One of the biggest concerns with cloud data storage are the data confidentiality and integrity verification at untrusted servers at server side. This is for saving money and storage space the service provider might delete rarely accessed files which belong to an normal client. So, the clients need that their data remain secure over the CSP.

Encrypting the sensitive data before outsourcing to servers using cryptographic schemes can handle the issue of confidentiality. However, the integrity of clients data in the cloud may be at risk due to the following reasons: The conventional cryptographic primitives for data integrity and availability based on hashing and schemes are not applicable on the outsourced data without having a copy of the data.

In order to solve the problem of data integrity, many schemes are proposed under different schemes and security models [2], [3], [4], [5], [6], [7], [11]. Mostly the model,categories: private auditability and public auditability. The private auditability achieves higher efficiency and public verifiability allows Third party Auditor (TPA) instead of client (data owner) to challenge the cloud server for correctness of data storage while not keeping private information.

Although the existing schemes aim to provide integrity verification for different data storage systems, the problem of supporting both public verifiability and data dynamics has not been fully addressed.

Recently, Wang et al.[15] proposed Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing using Merkle hash tree but they does not consider the problem of data confidentiality, which is one of the important issue for cloud data storage.

2. Related Work

Recently, many of growing interest has been pursued in the context of remotely stored data verification [2], [3], [4], [5],[6], [7], [8], [9], [10], [13], [14],[15][16]. Ateniese et al. [2] are the first one to consider public auditability in "provable data possession" model for ensuring possession of files on untrusted storages. In their scheme, they utilize RSA-based homomorphic tags for auditing of outsourced data. In their work [12] Ateniese et al. propose a dynamic version of the prior PDP scheme. However, the system imposes a priori bound on the number of queries and doesn't support for fully dynamic data operations i.e. it only allows only basic block operations with limited functionality and block insertions are not supported. In Enabling Public Auditabity and Data Dynamics for Storage Security in Cloud Computing[13] Wang et al. consider dynamic data storage in a distributed scenario and the proposed challenge is responsible for both to determine the data correctness and to locate possible errors. In addition, public auditability is not supported in their scheme. Shacham and Waters [4] designed an improved PoR scheme with lots of proofs of security in the security model defined in [3]. They use publicly verifiable homomorphic authenticators built from BLS signatures [16] which is based on the proofs which can be aggregated into a small authenticator value and public retrievability is achieved. Still, the authors consider only static data files. Erway et al. [14] were the first to implement the constructions for dynamic provable data possession. They enhance the PDP model in [2] 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.

All the existing schemes are aim at providing integrity verification for different data storage systems, the problem of supporting both public verifiability and data dynamics has not been fully addressed.

Recently, Wang et al.[15] proposed Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing using Merkle hash tree. Their scheme achieved the integrity of data in cloud with support of public verifiability and dynamic data operations. However, their scheme donot consider the confidentiality of data stored in cloud.

3. Formulation and System Model

3.1 System Model

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

User: who stores the data in the cloud that can be either enterprise or individual customers.

Cloud Server (CS): an entity which is managed by the cloud service provider (CSP) to provide data storage service having significant storage space and computation resources

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

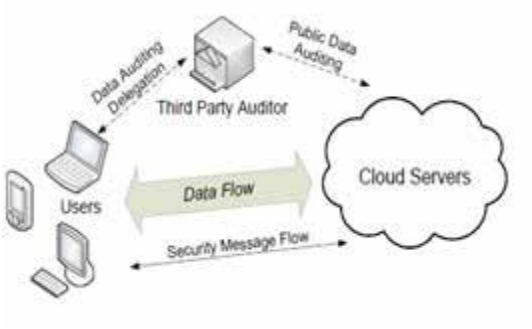

Fig.1 cloud storage architecture

In cloud data storage, a user stores his data through a CSP into a set of cloud servers, which are running in a Fig.1. Cloud Storage Service Architecture.

3.2 Threat Model

The threats can come from two different sources: internal attacks and external attacks. For internal attacks, a CSP can be self-interested, untrusted and possibly mal-icious. Not only it desire to move data that has not been or is rarely accessed to a lower tier of storage than agreed for monetary reasons, but it may also attempt to hide a data loss incident due to management errors, failures etc.

For external attacks, data integrity threats may comes from outsiders who are beyond the domain of CSP.

3.3 Design Goals

1. Storage integrity: to ensure users that their data are stored appropriately and kept intact all the time in the cloud.
2. Confidentiality: providing security for the data by using encryption technique such as RSA.

3.4 Preliminaries and notations

- F- the data file to be stored , which is denoted as a sequence of n blocks m1….mn ε Zq for some large prime q.
- fkey(.)- PseudoRandom Function (PRF) defined as: {0,1}*× key→Zq.
- πkey(.)-PseudoRandom Permutation (PRP) defined as:{0,1}*× key→{0,1}log2(n).
- H1(.),H2(.)→map to point collision-free hash functions defined as:{0,1}*→G, where G is a group.

Bilinear Map:-let G1, G2, and G be multiplicative cyclic groups of prime order q and let g1 and g2 be generators of G1 and G2 respectively and e be a bilinear map if e:G1×G2→G is a map with following properties:

Computable:There is a polynomial computable time algorithm to compute e(u,v) ε G1 for any (u, v) ε G.

Bilinear:For all $u \in G_1$, $v \in G_2$ and $x, y \in Z_q$

$$e(u^x, u^y) = e(u,v)^{xy} \quad (1)$$

Non-degenarate: If g1 is a generators of G1 and g2 is a genarators of G2, then

$$e(g_1, g_2) \neq 1 \quad (2)$$

For any $u_1, u_2 \in G_1$, $v \in G_2$

$$e(u_1 u_2, v) = e(u_1, v)\, e(u_2, v) \quad (3)$$

Merkle Hash Tree (MHT): A Merkle Hash Tree(MHT) is a data structure[17], which is used to prove efficiently and securely that a set of elements are not damaged and not altered. It is binary search tree, where each of the leaf node contains hash value of authenticated data. While MHT is commonly used to authenticate the hash values of data blocks however, in this we further employ MHT to authenticate both their values and the positions of data blocks and compute the root in MHT.

Definitions

The proposed scheme follows: KeyGen(1k)→(pk,sk)- is a random key generation algorithm that is run by the client to setup the scheme which takes a large security parameter k as input and produces a public/private key pair (pk,pr) based on RSA .

Enc (F)→F'. The Client uses this algorithm to encrypt the unprocessed file F with the seal key ek and encode it.

SigGen(pk,sk,m)→ σ_i is a(possibly random) algorithm run by client to generate verification of metadata which are signatures. It takes public key pk, secrete key sk and file block(m) as inputs and produce metadata as output i.e σ_i .

GenProof(pk,F',Q,Ø)→P is run by cloud server to generate integrity proof of data storage. It takes public key pk, file F', signatures Ø, and challenge query Q as inputs and produce output P, where P= (μ, σ)

VerifyProof(pk,chal,P)→{0,1} - This algorithm runs by TPA to validate the Proof of integrity from cloud server which takes public key pk, challenge query Q , and proof P and return output as 1 in case of the integrity of file is verified as correct otherwise 0.

ExecUpdate(F',Ø,update)→(F'',Ø',Pupdate).This algorithm run by server, it takes file F', signatures Ø, and a data operation request "update" form user and produce updated file F'', updated signatures Ø and a Proof Pupdate for the operation.

VerifyUpdate(pk,update,Pupdate)→this algorithm run by TPA. It takes public key pk, an operation request "update" and proof Pupdate from server if the verification success, it outputs 1 otherwise 0.

RSA Algorithm:

The RSA algorithm involves three steps: key generation, encryption and decryption.

Key generation

RSA involves a public key and a private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.

4. Proposed System

To ensure the integrity and Confidentiality of data storage we propose a new data verification protocol. It is designed based on merical hash functions. Our protocol consists of three phases: 1) Pre-processing 2) Data Integrity checking

4.1 Pre-Processing

The processing of this as follows: first, it generates the public and private key pair using Keygen function and then client encrypts the file using encryption function. After encrypting the data file, it computes the signatures over the encrypted file using SinGen algorithm. Finally, the client sends Signatures and public key TPA. Algorithm of Pre-processing is

Algorithm 1: Pre-Processing α

1 . Client generate private key : g, g

2. generate public key gβ
3. Encript the file {F} g $g^{\alpha\beta}$
4. Create the root for MHT . {H(R)} $g^{\alpha\beta}$,{dk}$g^{\alpha\beta}$
5. Sign the root sig(H(R))=H(I
6. Compute Signature f'{m'^{i}}

$\phi = \{\sigma_i\}$ where 1<i<N

$$\sigma_i = [H(m_i)^{\mu m}]^{\beta}$$

4.2 Verification phase

In this the TPA verifies the integrity of data as follows: first, the TPA creates a challenge and sends the server. After receiving a challenge, the server computes response and returns back to the TPA then TPA checks the integrity comparing response with signatures.

The detailed verification phase is given in algorithm 2.

Algorithm 2 verification Phase

1: Procedure
2: generate the challenge query Q={i,Vi}
3. send {Q, pk}iεI to Server
4: for each query server calculates

$$\sigma = \Pi_{i\in I}\sigma_i^{V_i} \in G_1$$

And

$$\mu = \sum_{i\in I} V_i m_i$$

$$where\ \sigma_i = (H_1(i)\ u^{m_i})^x \in G_1 (i = 1.....,n)$$

5: server sends {F',σ,T,} to TPA

$$if\ (e(\sigma \cdot (T^{H_2(T)}, g) = e(\prod_{i=s_1}^{s_l} H_1(i)^{V_i} \cdot u^{\mu}, v))$$

6: TPA verify
7. return 1
8. else
9. return 0
10. end if
11. end procedure

5. Security Analysis

In this section, we analyze that proposed scheme is more secure than existing schemes against data leakage and data loss/damage. Our proof consists of two parts: Integrity and Confidentiality.

a)Integrity of data storage guaranty:

We need to prove that cloud server cannot generate valid response towards TPA without faithfully storing the data file.

Theorem 1. If cloud server passes the verification phase, then it must indeed possess the particular data stored correctly or not.

Proof. This theorem consists of three steps

1. Initially we show that there exists no server that can forge a valid response {σ, μ, T} to pass the verification using equation (1). The integrity of this statement follows from theorem available in 4.2[8]
2. Now, we show that if the response from {σ, μ, T} is valid, where μ=μ'+tH2(T) and T=(v2)t, then the important sample blocks in μ' must be valid. This is obtained immediately by verifying the response using equation (2)
3. Finally, in our scheme should detect all data corruptions if data has been corrupted during the verification phase. Assume that if server corrupts some of the blocks {mij}, in μ', where μ'=m'j. This is achieved by checking response with previously computed signatures using equation (3). If not. It indicates that data has been corrupted.

Theorem 2: The stored data cannot be leaked to unauthorized parties

Proof: We prove that theorem in two steps: first, we show that no information on μ' can be learned from μ, this is because the file is encrypted by using RSA-algorithm[19] where p and q values are chosen randomly and large. If attacker try to access a encrypted file, he need private key. If tries to get the private key by using public, however, it is impossible due RSA assumption. Therefore, according to our analysis, an adversary cannot get anything from encrypted file. Hence, it is proved that proposed scheme is more secure against data leakage.

6. Performance Analysis

According to the algorithm 1 and 2, we can demonstrate the overall workload of the computing and storage of each parties in our scheme as followed:

- Client: who stores the private key α and decryption key dk of file; computes the public key β and the signature Sigsk(H(R)) of the MHT root H(R).
- TPA: stores the user's public key g, encrypts/decrypts the file, computes the data blocks signature collection Φ, and verifies the both the equations during verification.
- CSP: stores the signature Sigsk(H(R)), the encoded F' and the Φ; generates the verification information μ and ω and then computes the Ωi for recovering the MHT.

7. Conclusion

This approach is very secure. In order to attain data integrity we are verifying through are using Third Party Auditor (TPA) on behalf of cloud client to verify the integrity of data storage using Merical Hash Tree i.e all the verification is done by using TPA only which reduces the cost as well as users burden and thus providing cost effective method to the user. The confidentiality is attain by using RSA based cryptography algorithm which computes very fastly. In this, the out sourced data is encrypted with public key and made into cipher text and at the client side the user decrypt the encoded data using private key. Hence providing integrity and confidentiality to the user's data.

REFERENCES

[1]. 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 ComputerSecurity (ESORICS '09), pp. 355-370, 2009. | [2] G. Ateniese, R. Burns, R. Curtmola, J.Herring, L. Kissner, Z.Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security (CCS'07), pp. 598-609, 2007. | [3] A. Juels and B.S. Kaliski Jr., "Pors: Proofs of Retrievability for Large Files," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 584-597, 2007. | [4] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '08), pp. 90-107, 2008. | [5] K.D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Report 2008/175, Cryptology ePrintArchive, 2008. | [6] M. Naor and G.N. Rothblum, "The Complexity of Online Memory Checking," Proc. 46th Ann. IEEE Symp. Foundations of Computer Science (FOCS '05), pp. 573-584, 2005. | [7] E.-C. Chang and J. Xu, "Remote Integrity Check with Dishonest Storage Server," Proc. 13th European Symp. Research in ComputerSecurity (ESORICS '08), pp. 223-237, 2008. | [8] M.A. Shah, R. Swaminathan, and M. Baker, "Privacy-Preserving Audit and Extraction of Digital Contents," Report 2008/186, Cryptology ePrint Archive, 2008. | [9] A. Oprea, M.K. Reiter, and K. Yang, "Space-Efficient Block Storage Integrity," Proc. 12th Ann. Network and Distributed System Security Symp. (NDSS '05), 2005. | [10] T. Schwarz and E.L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage,"Proc. 26th IEEE Int'l Conf. Distributed Computing Systems (ICDCS'06), p. 12, 2006. | [11] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," Proc.IEEE INFOCOM, pp. 954-962, Apr. 2009. | [12] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. Fourth Int'l Conf. Security and Privacy in Comm. Networks (SecureComm '08), pp. 1-10,2008. | [13] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring Data Storage Security in Cloud Computing," Proc. 17th Int'l Workshop Quality of Service (IWQoS '09), 2009. | [14] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), 2009. | [15] Q. Wang, C. Wang, and K. Ren are with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing" | [16] D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing," Proc. Seventh Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT'01), pp. 514-532, 2001. | [17] R.C. Merkle, "Protocols for Public Key Cryptosystems," Proc. IEEE Symp. Security and Privacy, pp. 122-133, 1980. | [18] K.D. Bowers, A. Juels, and A. Oprea, "Hail: A High-Availability and Integrity Layer for Cloud Storage," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), pp. 187-198, 2009. | [19] H.-M. Sun and M.-E.Wu, An approach towards Rebalanced RSA-CRT with short public exponent Cryptology ePrint Archive, Report 2005/053, 2005 [Online]. Available: http://eprint.iacr.org/2005/053 |