INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

CLOUD SECURITY THROUGH DELEGATED TPA

SWATI S. KACHARE, Dr. P. K. DESHMUKH

Rajarshee Shahu, College of Engg., Pune

Accepted Date:

27/02/2013

Publish Date:

01/04/2013

Keywords

Cloud Computing,

Data Integrity,

confidentially,

Third Party Auditator,

ECC,

Sobol Sequence.

Corresponding Author

Ms. Swati S. Kachare

Abstract

Users store any kind of data into cloud and provide the demand of long term continuous assurance of their stored data safely. Data integrity of the remotely stored data has becomes more challenging because users can't have direct control on their own data. Our design is based on Elliptic Curve Cryptography and Sobol sequence. Here allows third party auditor to periodically verify the data integrity stored at CSP without retrieving original data. It generates probabilistic proofs of integrity by challenging random sets of blocks from the server. Most importantly, our protocol is confidential: it never reveals the data contents to the malicious parties. It removes burden of both the user's and storage service's fear about data leakage and data corruptions.

INTRODUCTION

Cloud computing is an internet based computer technology so users can store any kind of data like photos, videos, favourite movies etc. Some of the major firms like Amazon, Microsoft and Google have implemented cloud to speed up their business. Cloud is pool of computing services on a large scale. Storage of large amount of data into cloud reduces cost and maintenance but the end users or customers may not know where the data stored. They lose his control over data. So the necessary thing is to ensure the data from modification or alteration during the period of storage. Cloud computing also brings in many new security challenges on protecting the integrity and privacy of user's data.

Since Ateniese [1] and Juels [7], several works aim to provide a method to efficiently verify the integrity of user's data capabilities. The client has not physical possession of data for that they are facing risk for missing or correcupted data. If Cloud Storage Service (CSS) behaves like vulnerable to attacks or failures which not only manage the stored data but also maintain it. Because of irretrievable act of it, user losses data. Since their data or archives are stored into an uncertain storage pool outside the enterprises.TPA is the third party auditor who will audit the data of data owner or client, released audit report helps the owner or client to evaluate the risk of their subscribed cloud data services and also it will be beneficial for the cloud. Behaviours of CSS not are known by the cloud users, even if this dispute may result from the users' own improper operations [4]. Therefore, it is necessary for cloud service providers to offer an efficient audit service to check the integrity and availability of the stored data [5].To audit the correctness of data cloud environment can be formidable and expensive for the cloud users [6].It is crucial to realize public auditability for CSS so data owner may restore to third party auditor (TPA) for who has expertise and capabilities.The notions of proof of retrievability (POR) [7] and provable data possession (PDP) [8] have been proposed to implement public audit ability. Drawbacks like information leakage of verified data in verification affect the impact of cloud audit

service. Thus new models are needed to enable the security of public verification protocol. Although the existing schemes aim to provide integrity verification for different data storage systems, but problem of confidentiality of data has not been fully addressed. The protocols [8][9] verifying integrity of outsourced data based on pseudorandom sequence, which does not cover the whole data while computing the integrity proof, it have been proposed to ensure the confidentiality and integrity of remote data. But, all these schemes are unable to provide strong security assurance to the users.
Our contribution can be summarized as follows: Our method allows efficient audit services for outsourced data. We have proposed audit system model for verifying integrity of outsourced data storage. This scheme retains soundness and completeness property. Here, we have also proposed two process model which consist setup and verification phases. Probabilistic proofs of integrity by challenging random sets of blocks from the server, which drastically reduces the communication and I/O costs.

Section 2 describes the related work. Section 3 introduces proposed scheme which further includes Audit system model, Audit system construction. Security Analysis is given in section 4.We introduces conclusion in section 5

II. RELATED WORK

There are various mechanism proposes for how to use the TPA so that it relieves the burden of data owner of local data storage and maintenance, it also eliminates their physical control of storage dependability and security, which traditionally has been expected by both enterprises and individuals with high service-level requirements. Such an auditing service not only helps save data owners 'computation resources but also provides a transparent yet cost-effective method for data owners to gain trust in the cloud. Ateniese et al. [1] are the first who propose public auditability model for ensuring possession of files on less trusted storage. A straightforward approach like message authentication codes (MACs) can be use to protect the data integrity.User can verify the integrity by recalculating the MAC of the received data file and comparing it to the locally recomputed value. If the data file is large, MACs cannot be employed. Hash tree does not give any assurance about the correctness of other outsourced data where the leaves are hashes of data blocks and internal nodes are hashes of their children of the tree. The data owner only needs to store the root hash of the tree to authenticate his received. So TPA can be used who performs this thing for data owner. Cong Wang et al propose cloud security while using TPA [2] where basic scheme eliminates the involvement of the client which is indeed intact. This method saves the computational resource and cost of storage of data owner but how to trust on TPA are not calculated. If TPA become intruder and pass information of data owner to unauthorized user or TPA also modifying a data or deleting a data than how owner know about this problem are not solved.

Audit system architecture [10] describes the leakage of data and prevents the fraudulence prover. Elliptic Curve Cryptography and Sobol Sequence [11] allows third party auditor to periodically verify the data integrity stored at CSP without retrieving original data.

III. PROPOSED SCHEME

1. Audit System Architecture/Model

The Audit System model is illustrated in Fig.1. which consist the following things:

1) Cloud User: The cloud user may be an individual or an organization that can originally store their data in cloud and access that data.

2) Cloud Service Provider (CSP): Who manages cloud servers (CSs) and provides a paid storage space on its infrastructure to users as a service.

3)Third Party Auditor (TPA) or Verifier: The TPA or Verifier, who has expertise and capabilities that users may not have and verifies the integrity of outsourced data in cloud on behalf of users. Based on the audit result, which release an audit report to user.

The given Model is known as audit service outsourcing due to data integrity verification where user or data owner need to dynamically interact to CSP to access or update data. Here TPA is used as Third Trust Party to ensure the storage security of the outsourced data.

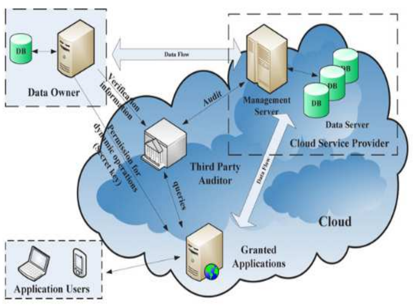


Fig. 1. Audit system architecture for cloud computing.

2. Audit System Construction

We have proposed Efficient and secure scheme to support our audit system in cloud which ensure confidentially and integrity; we have used Elliptic Curve Cryptography. This Protocol consist of two phases namely Setup and Verification which is depicted in fig.2.

2.1 Setup:

In the Setup phase the user pre-processes the file before storing in cloud which consists of three algorithms: KeyGen

Encryption , MetadataGen.

2.1.1 KeyGen

In this algorithm user takes k as input and generate public key *PK= {b, n, P}* and private key PR= {N_m}.The given security parameter *k* (k>512), user chooses two large primes p and q of size k such that p=q=2 (mod 3). Then compute

$$\mathbf{n= pq} \qquad (1)$$

$$\mathbf{N_m= LCM\ (p+1,\ q+1)} \qquad (2)$$

Where $\mathbf{N_m}$ is order of ellipses curve over ring Z_n denoted by En (o, b) and b is randomly chosen integer number. Algorithm for KeyGen:

1. **Procedure**: KeyGen (k) ← *{PK, PR}*

2. Take security parameter k (k>512)

3. Choose two random primes p and q of size k: p=q= 2 (mod 3)

4. Compute **n= pq**

5. Compute $\mathbf{N_m}$ **= LCM (p+1, q+1)**

6. Generate random integer b<n, gcd (b, n) =1

7. Compute P is a generator of En (0, b)

8. Private Key PR= {Nm}

9. Public key PK= {n, b, P}

10. End

2.1.2 Encryption

To ensure the confidentiality of data, the user encrypts each data block *m* in the file *F.* It takes input as m_i is keyed Sobol Random Function (SRF) and secrete random parameter *s* and produce $\boldsymbol{m'_i}$ as output.

$$\mathbf{F = \{m_1, m_2, \ldots\ldots\ldots, m_n\} = \{m_i\}_{1<i<n}} \qquad (3)$$

$$\mathbf{F' = m'_i = m_i + f_k(s)} \qquad (4)$$

Algorithm for Encryption:

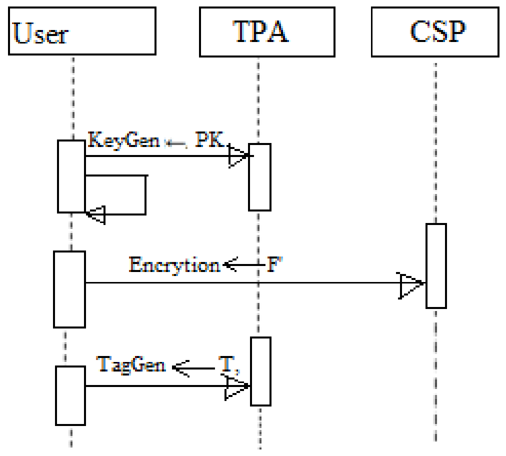
```
1. Procedure: Encryption ($m_i$ , s) $\leftarrow m'_i$
2. for 1 to n
3. Compute $m'i = m_i + f_k(s)$
4. End
```

2.1.3 MetadataGen:

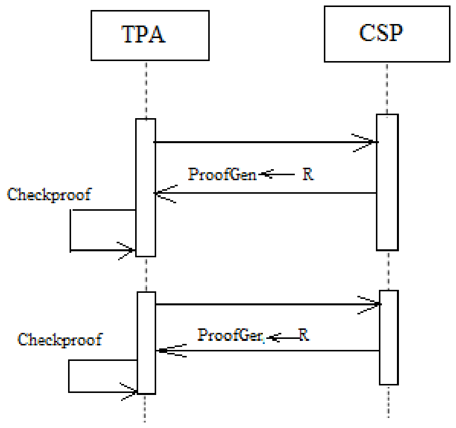
After encrypting the data, the user computes a metadata over encrypted data to verify the integrity of data which takes m'_i, public key and private key as inputs and produce metadata **Mdi** as output:

$$\mathbf{Mdi} \leftarrow m'_i\, P\ (mod\ \ N_m) \qquad (5)$$

After computation of metadata, the user sends metadata, PK to the TPA for verification and sends file *F' to* cloud servers for storage.



(a)SetupPhase



(a) Verification phase

2.2 Verification Phase

To verify the integrity of data, the verifier first creates a challenge and sends to the server. After receiving a challenge from the verifier, the server computes a response as integrity proof and return to the verifier. It consists of three algorithms: Challenge, ProofGen and CheckProof.

2.2.1. Challenge

The verifier creates a challenge by using k_{SRF}, j and Q as input and return *chal* as output. The verifier chooses a random key k_{SRF} and k_{SRP} using Sobol sequence and compute random indices $1<i_j<c<n$ (j=1......c) Where $c=\prod k_{SRP}$ (6) this prevents the server from anticipating, in which blocks will be queried in each challenge. The verifier also generates a fresh random value **v** to guarantee that the server does not reuse any values from the previous challenge and computes $Q=\mathbf{v}P$ (7) Then, verifier creates the challenge and sends to the server as Chal = {k, j, Q}

2.2.2.ProofGen

Server computes a response R as integrity proof. It takes inputs as encrypted data m'_i, challenge *chal* .First; it generates random numbers using Sobol random Function (SRF) i.e. $a_j=f_{k_{SRF}}(j)$ (8)

Then compute $b=\sum_{j=1}^{c} a_j\, m'_{ij}$ (9)

$R = bQ \bmod n$ (10)

$= \sum_{j=1}^{c} a_j\, m'_{ij}\, Q \bmod n$

$= \sum_{j=1}^{c} a_j\, m'_{ij}\, rP \bmod n$

$= r\,(\sum_{j=1}^{c} a_j\, m'_{ij}\, P \bmod n)$

2.2.3. CheckProof

After receiving a response from the server, the Verifier checks the integrity. It takes public key *pk*, challenge query *chal*, and proof *R* as inputs and return output as 1 if the integrity of file is verified as successfully or 0 as follows: the verifier re-generates random numbers using Sobol Random function i.e. $a_j = f_{SRF}(j)$. Then compute

$$s = \prod_{j=1}^{c} a_j \, Md_i' \bmod n \qquad (11)$$

$$R' = rS \bmod n \qquad (12)$$

verifier checks if response is valid,

$$R' = R \qquad (13)$$

Then it returns 1 otherwise 0.

IV. SECURITY ANALYSIS

In this section, we present the formal security analysis of integrity and confidentiality of data stored in cloud.

4.1 Integrity

We have depended on the Finding order of elliptic curve and Elliptic curve discrete logarithm problem denoted by ELDL problems. Here, we show that our protocol is sound against dishonest server based on previous transactions and pre-computed metadata. There are four possibilities that the server can compute the integrity proof without storing the user's data:

1)The server use pre-computed value. However, guessing occurs with negligible probability and pre-computing the correct response is not possible because each time the verifier challenge the server with a fresh challenge.

2)The server replayed the previous response. In this case, the server would have to find r from challenge ***chal*** to compute the correct proof. Since *r* is chosen randomly, finding *r* is hard based on ELDL problem.

3) The server has an algorithm to compute *m'i mod N* with inputs instead of storing *m'in* [1=i=n]. But this option is not possible, because, the server cannot compute N based on the hardness of solving the order of elliptic curve En (0, b).

4) Last option for server is, if it does not store the data {m'i} and it may try to collude with the other servers for storing the same data. However, this option is not feasible, since data stored at each server is securely encrypted using Sobol Random Function (SRF). The *f* is a keyed one-way function and *s* is a secrete parameter, so, no one except the user can retrieve the original data m_i from $m'_{i.}$ All these options lead to contradiction; so the server cannot

compute response without storing the data. Hence, our protocol is complete.

We are proving ***the proposed protocol is complete*** according to the sound and commutative property of point multiplication in an elliptic curve [15].

We have R '=R and *R* ' =*rS* mod n

$$S = \prod_{j=1}^{c} a_j Md'i \bmod n$$

$$= \prod_{j=1}^{c} (a_j m'_{ij\,P} \bmod N_m) \bmod n$$

$$= \sum_{j=1}^{c} a_j\ m'_{ij}\ P \bmod n$$

R' = vS mod n

$$= v \prod_{j=1}^{c} (a_j m'_{ij} P \bmod n)$$

$$= v \left(\sum_{j=1}^{c} a_j\ m'_{ij} P \bmod n\right)$$

$$= R$$

From the equation (13), the protocol is **complete or valid.** Then the verifier is "probabilistically" assured that server still holds data safely.

4.2 Probability Detection

After receiving a response from the server, the verifier check whether response is valid or not? If it is not valid, then the verifier detects the corruptions with high probability .Here, we investigate how the probabilistic nature of the proposed protocol makes it possible to ensure integrity. We are making the following assumptions:

● The verifier's random selection of indexes is uniform i.e., for n blocks, the probability to pick any block is 1/n.

● If an attacker removes a portion d/n of data from the storage file; this portion is referred to as the corruption of the file.

●The verifier performs on average c challenges: 1= c =n (n is the number of blocks) and detects the corruption with high probability. The detection probability PD of disrupted blocks is an important parameter to guarantee that these blocks can be modified or detected in a time. We have detection of probability is:

PD=1-(1-d/n) c

For a given probability of detection of a data corruption, it is possible to probabilistically determine the average number of challenges that the verifier should perform to achieve the probability of detection. The number of challenges *c* can be derived as follows: C=$\log_{1\ d/n}$ (1-PD)

3.2 Confidentiality

In this analysis, we depend on the hardness of the Elliptive Curve Diffie Hellman (ECHP) and Elliptive Curve Discrete Logarithm (ECDL) problems. We prove ***Data leakage to attacker is prevented by the proposed protocol*** under different attacks: 1) By communication link between the user and server the secret parameter *s* cannot be derived because of Elliptive Curve Diffie-Hellman (ECDH) problem. The public parameter cannot help to calculate any useful information that can reveal the shared key between the user and server. 2) Suppose, If the corrupted server wants to access the data from the encrypted file F'=m'. It's not possible, because he should need a secrete parameter, which is chosen randomly by user. Due to the ECDL problem, to get the secret key by using different combinations of public parameters server fail to do this. Hence, the server cannot learn anything from F'.
basis of ECDH and ECDL problems; our proposed scheme is confidential against data leakage.

*V.*CONCLUSION

In this paper, we proposed an efficient and secure protocol by using ECC and Sobol sequence to address the construction of an efficient audit system for data integrity in clouds. The proposed scheme supports public verifiability that enables TPA to verify the integrity of data without retrieving original data from the server and probability detects data corruptions. Moreover, this scheme secures in terms of integrity and confidentiality through security analysis and archive the detection of servers misbehaviour with a high probability.

REFERENCES

1. G.Ateniese et al., Provable Data Possession at Untrusted Stores, Proc. ACM CCS =07, Oct. 2007, pp. 598–609.

2. Cong Wang and Kui Ren, Wenjing Lou, Jin Li, toward Publicly Auditable Secure Cloud Data Storage Services in IEEE Network July/August 2010

3. Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring,Lea Kissner, Zachary Peterson "Provable Data Possession at untrusted stores",ACM, 2007.

4. Ko, R.K.L., Lee, B.S., Pearson, S., 2011. Towards achieving accountability, auditable- ity and trust in cloud computing. Vol.193 of Communications in Computer and Information Science. Springer, Berlin/Heidelberg, pp. 432–444.

5. Yavuz, A.A., Ning, P., 2009. Baf: An efficient publicly variable secure audit logging scheme for distributed systems. In: ACSAC, pp.219–228.

6. Armbrust,M.,Fox,A.,Griffith,R.,Joseph,A. D.,Katz,R.H.,Konwinski,Patterson,D.A., Rabkin, A., Stoica, I., Zaharia, M., 2010.Aview of cloud computing.Commun. ACM 53 (4), 50–58.

7. Juels Jr., A., Kaliski, and B.S., 2007.Pors: proofs of retrievability for large files. In: Proceedings `Ofthe 2007 ACM Conference on Computer

8. M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," *Proc. 11th USENIX Workshop on Hot Topics in Operating Systems (HOTOS '07)*, 2007, pp. 1–6, CA, USA.

9. L. Chen, G. Guo, "An Efficient Remote Data Possession Checking in Cloud Storage", International Journal of Digital Content Technology and its Applications. Volume 5, April 2011, pp.43-50.

10. Yan Zhu, Hongxin Hu, Gail-Joon Ahn, Stephen S. Yu,"Efficient Audit service Outsourcing for data integrity in the clouds", The Journal of systems and Software.85(2012).

11. Syam Kumar P, Subramanian R,” An Efficient and Secure Protocol for Ensuring Data Storage Security in Cloud Computing”, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 1, Nov 2011.

12. Abhishek Mohta, Ravi Kant Sahu,, Lalit Kumar Awasthi,” Robust Data Security for Cloud while using Third Party Auditor”, International Journal of Advanced Research in Computer Science and Software Engineering.

13. K. Koyama, U. Maurer, T. Okamoto, and S. Vanstone, “New Public-Key Schemes Based on Elliptic Curves over the Ring Zn”, Advances in Cryptology - CRYPTO '91, Lecture Notes in Computer Science, Springer-Verlag, vol. 576, Aug 1991, pp. 252-266,.