Ensuring Data Storage Security in Cloud Computing using Sobol Sequence

P. Syam Kumar, R. Subramanian and D. Thamizh Selvam
Department of Computer Science, School of Engineering and Technology, Pondicherry University, Puducherry. India
shyam.553@gmail.com, rsmanian.csc@pondiuni.edu.in and dthamizhselvam@gmail.com

Abstract— **Cloud computing is the next stage in evolution of the internet, which provides large amount of computing and storage to customers provisioned as a service over the internet. However, cloud computing facing so many security challenges due to the possible compromise or byzantine failures. In this paper, we focus on Ensuring data storage security in cloud computing, which is an important aspect of Quality of Service (QoS). We propose an effective and flexible distribution verification protocol to address data storage security in cloud computing. In this protocol, we rely on erasure code for the availability, reliability of data and utilize token pre-computation using Sobol Sequence to verify the integrity of erasure coded data rather than Pseudorandom Data in existing system. Unlike prior works, our scheme provides more security to user data stored in cloud computing. The performance analysis shows that our scheme is more secure than existing system against Byzantine failure, unauthorized data modification attacks, and even cloud server colluding attacks.**

Keyword: Cloud computing; Data storage security; Availability; Reliability; Integrity; Pseudorandom data; Sobol sequence

I. INTRODUCTION

Cloud computing is an internet based computing. It dynamically delivers everything as a service over the internet based on user demand, such as network, operating system, storage, hardware, software, and resources. These services are classified into three types: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). Cloud computing is deployed as three models such as Public, Private, and Hybrid clouds [1].

Cloud data storage (Storage as a Service) is an important service of cloud computing referred as Infrastructure as a Service (IaaS).

Data storage in cloud offers so many benefits to users:

1) It provides unlimited data storage space for storing user's data.
2) Users can access the data from the cloud provider via internet anywhere in the world not on a single machine.
3) We do not buy any storage device for storing our data and have no responsibility for local machines to maintain data.

Amazon's Elastic Compute Cloud (EC^2) and Amazon Simple Storage Service (S3) ([2], [12]) are well known examples of cloud data storage. On the other side along with these benefits' cloud computing faces big challenge i.e. data storage security problem, which is an important aspect of Quality of Service (QoS). Once user puts data on the cloud rather than locally, he has no control over it i.e. unauthorized users could modify user's data or destroy it and even cloud server collude attacks. Cloud users are mostly worried about the security and reliability of their data in the cloud. Amazon's S3 [3] is such a good example.

The following research works have been highlighted the importance of ensuring integrity and availability of outsourced data. POR schemes ([4]-[6]) efficiently verifies the server for outsourced data storage correctness. G.Atenies et al. [7] introduced a Provable Data Possession (PDP) scheme; it efficiently detects the large no of file corruptions. In addition to the PDP scheme, R.D.Pietro et al. [8] introduced a Scalable Data Possession (SDP) to verify the integrity of remotely stored data dynamically. However, all the schemes could't addresses all security threats in cloud data storage, since they work only for single server.

In the complementary approach, researchers also proposed a distributed protocols ([9]-[11]) to verify the data storage security on multiple servers, but these schemes will not addresses all security issues of cloud data storage.

Recently, Wang et al. [12] proposed a homomorpic distributed verification protocol to ensure data storage security in cloud computing using Pseudorandom Data. Their scheme achieves the storage correctness as well as identifies misbehaving servers. However, this scheme was not providing full protection for user storage data in cloud computing, because Pseudorandom Data does not cover the entire data while verifying the cloud servers for data storage correctness i.e. some data corruptions may be missing.

In this paper, we propose a distributed verification protocol to guaranty the data storage security in cloud computing. This scheme uses Reed-Solomon erasure code [13] for the availability and reliability of data and utilizes the token pre-computation using Sobol Sequence[14] rather than Pseudorandom Data [12] to check the integrity of erasure coded data in cloud data storage. Our method achieves the integrity of storage correctness guaranty and identification of misbehaving servers i.e. whenever data modifications or deletions have been detected during the storage correctness verification across cloud servers, this method should guarantee the identification of misbehaving servers. Compared to previous methods, our method should provide more security to users data stored in cloud computing. Figure 1 shows the comparison of Pseudorandom Data and Sobol Sequence.

978-1-4244-7674-9/10/$26.00 ©2010 IEEE

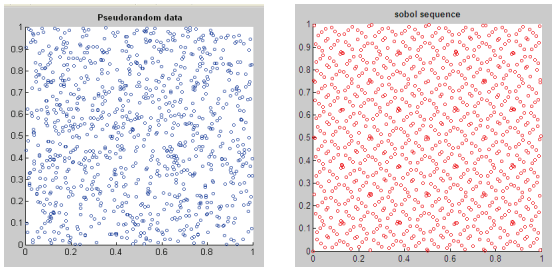

Figure.1 Comparison of Pseudorandom Data and Sobol Sequence.

The rest of the paper is organized as follows: Section **2** describes the related work, section **3** introduces problem definition. Section **4** provides detail description about our proposed model. Section **5** gives performance evaluation of our model and Section. **6** give the conclusion of our work.

II. Related Work

Ari Juels et al. [4] described a Proofs' of Retrievability (POR) model to ensure the outsourced data security. This scheme efficiently detects data corruptions and, achieves the guaranty of file retrievabilty. Shacham et al. [5] introduced a new model of POR, which enables unlimited no of queries for public verifiability with less overhead. Kennadi D et al. [6] proposed a theoretical framework for the design of POR. It improves the JK [4] and SW [5] models. All the schemes produce weak security, because they work only for single server. Later, in their subsequent work Kennadi Brow et al. [11] introduced a HAIL protocol, which extended the POR schemes on multiple servers. HAIL achieves the integrity and availability of data in cloud. However, this protocol will not address all the data security threats. Ateniese et al. [7] described a Provable Data Possession (PDP) to verify the integrity of outsourced data; it detects the large fraction of file corruption, but no guaranty of file retrivability. In their subsequent work R.D.Pietro et al. [8] proposed a Scalable Data Possession (SDP), this scheme overcomes all problems in the PDP scheme [7], but this scheme also works only for single server. Later, Curtomola et al. [17] described a Multiple Replica-Provable Data Possession (MR-PDP), which is an extension of PDP to ensure data availability and reliability of outsourced data on multiple servers. Compare to PDP [7], it requires only single set of tags to challenging servers.

In other existing works, Lillibridge et al. [10] proposed a new Internet based backup technique to store client data. It protects data from free riders and disrupter attacks. However, it can't detect data modifications or data changes. Schwarz et al. [9] presents a new model to check the security of data in distributed storage system. It verifies the large amount of data with minimum bandwidth in distributed storage systems. However, in this scheme server can access linear no of file blocks per each challenge. Fiho et al. [18] describes a secured hash function to prevent cheating in a P2P systems, however it is unusable when file is large. Shah et al. [19] proposed a new scheme, which allows Third Party Auditor (TPA) to keep on-line storage honesty with hash values computed by user on encrypted file. However, this scheme works only for encrypted files.

Recently, Wang et al. [12] described a homomorpic distributed verification scheme using Pseudorandom Data to verify the storage correctness of user data in cloud. This scheme achieves the guaranty of data availability, reliability, and integrity. However, this scheme was also not providing full protection to user data in cloud computing, since Pseudorandom Data would not cover the entire data.

Sobol Sequence [14] is an example of quasi-random low-discrepancy sequences. This sequence was first introduced by I.M.Sobol in 1967. The numbers are generated sequentially to fill the larger "gaps" in the Pseudorandom Data [12]

III. Problem Definition

A. *Systm Model*

The network representative architecture for cloud data storage, which contains three parts as shown in Figure 2, viz Users, Cloud Service Provider (CSP) and Third Party Auditor (TPA).

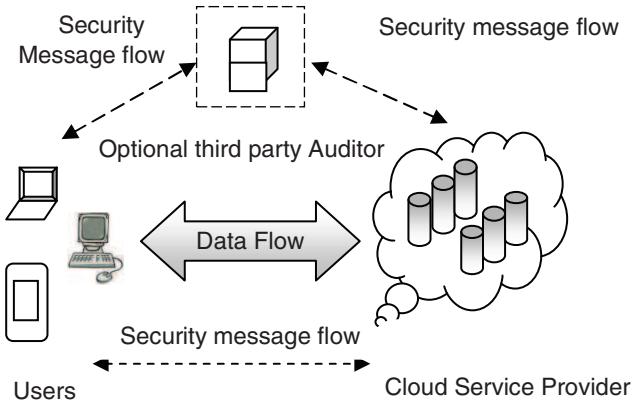

Figure. 2 Cloud Data Storage Architecture

as shown in Figure 2, the brief descriptions of these parts as follows:

Users: - Users who have data to be stored and interact with the cloud service provider (CSP) to manage their data on the cloud. They are typically, desktop computers, laptops, tablet computers, mobile phones, etc.

Cloud Service Provider (CSP):- Cloud service provider (CSP) has major resources and expertise in building and managing distributed cloud storage servers. A CSP offers storage or software services to user's available via the Internet.

Third Parity Auditor (TPA):- An optional TPA, who has expertise and capabilities that users may not have, is monitoring the risk of cloud data storage services on behalf of users.

To address the data storage security in cloud computing, we propose a distributed verification protocol using Sobol Sequence [14]. Our method allows a user to verify the cloud server whether data is stored correctly or not and identify the misbehaving servers without having a of local copy of data. In case users not have time to verify their data in cloud, they can assign this job to trusted Third party Auditor (TPA).

Our method main goal is to provide more security to user data stored in cloud computing i.e. guaranty the availability, reliability, and integrity of data and Users have to perform storage correctness checks with minimum overhead.

B. *Preliminaries and Notations*

- **F –** the data file to be stored, we assume that F can be denoted as a matrix of m equal-sized data vectors, each consisting of *l* blocks, these all data blocks belongs to Galois Field GF(2^P) where p= 8 or 16 or 32.
- **A–** The distributed matrix used for Reed-Solomon coding.
- **G –** is the encoded file matrix, which includes set of k=m+n vectors, each having *l* blocks.
- $f_{key}(.)$ – Sobol Random Function (SRF), which is defined as $f:\{0,1\}^* \times$key-GF(2^P).
- $\pi_{key}(.)$–Sobol Random Permutation (SRP) which is defined as $\Pi:\{0,1\}^{\log2(l)} \times$key– $\{0,1\}^{\log2(l).}$
- *v*er –which records the times the blocks have been modified. Initially we assume ver is 0 for all data blocks.
- s_{ij}^{ver} – the seed for SRF, which depends on the file name, block index i, the server position j, and version number *ver*.

IV. ENSURING CLOUD DATA SECURITY

To ensure data storage security in cloud computing, we propose a new model. Our proposed model consists of three phases 1) File Distribution 2) Token Pre-Computation 3) Challenge Response Protocol.

A. *File Distributions*

Erasure codes [13] have been employed for fault tolerance and improved performance in distributed storage systems. In cloud data storage, our method rely on this technique to distribute the entire data file across a set of k=m+n cloud servers for data availability and reliability. A (m+n, n) Reed-Solomon erasure coding [13] can be used to generate n parity blocks from m data blocks in such a way that original data file can be reconstructed from any m out of m+n. i.e. the original file can survive from any failures of m+n servers without any data loss and with less space overhead.

Let **F**= $\{F_1,F_2,\ldots F_m\}$ and $F_i = (f_{1i}, f_{2i}\ldots..f_{li})^T$ (i ϵ {1……m}), where $l \le 2^p$-1. File distribution is a symmetric layout with parity vectors are achieved with the data distributed matrix '**A**' derived from Vander monde Matrix [13].

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{bmatrix}$$

where $\mathbf{x_j}$ (j ϵ {1……k} are distinct elements randomly picked from **GF (2^p)**.

After doing a sequence of row elementary row transformations, the new matrix A is:

$$A=(I/P) = \begin{pmatrix} 1 & 0 & \ldots.. & 0 & p11 & p12 & \ldots. & P1n \\ 0 & 1 & \ldots. & 0 & p21 & p22 & \ldots. & P2n \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ 0 & 0 & & 1 & pm1 & pm2 & \ldots. & pmn \end{pmatrix}$$

where **I** is a m × m Identity matrix and **P** is a parity generation matrix with size m×n. Note that A is derived from Vander monde matrix, thus it has property that any m out of m+n rows form an invertible matrix.

To obtain encoded file, we have to multiply **F** by **A** using Reed-Solomon code Basic equation that is:

$$\mathbf{G= F .A}[13] = (G^{(1)},G^{(2)},\ldots\ldots,G^{(m)},G^{(m+1)},\ldots..,G^{(k)}) = (F_1, F_2, F_m, G^{(m+1)},\ldots\ldots..G^{(k)}$$

where $G^{(j)} = (g_1^{(j)}, g_2^{(j)}, \ldots.., g_l^{(j)})^T$ (j ϵ {1, …, k})

The encoded file contains both original data vectors of F and $G^{(m+1)},\ldots\ldots..G^{(k)}$ parity vectors generated based on original data vectors F.

B. *Token Pre-Computation*

After encoded the file, in order to achieve the guaranty of storage correctness of cloud data storage and identifying misbehaving servers, our scheme entirely relies on the token pre-computation using Sobol Sequence [14]. The main idea is, before distributing the file across cloud servers, the user pre-computes a certain number of short verification tokens $V_i^{(j)}$ over individual data vector $G^{(j)}$. Each token covers a random set of data blocks. After, the completion of Pre-Computed tokens, user stores all tokens locally. Whenever user wants to check the storage correctness of data in the cloud, he challenges the cloud servers with a set of randomly generated data block indices. Upon receiving request from users, then each cloud server must compute a short "signature" over the corresponding random block indices and return to the user. For the data integrity to hold, the response values $R_i^{(j)}$ must match the corresponding tokens $V_i^{(j)}$ pre-computed by user. Mean while all other servers operate over the same set of random data block

indices. The requested response values for integrity check also valid for codeword determined by parity matrix P.

The token pre-computation function considering here belongs to a family of Universal Hash Function (UHF) ([12], [15]) chosen to protect homomorphic properties, which can be perfectly integrated with the verification of coded data ([9], [16]).

Algorithm 1 Token Pre-Computation

1: **Procedure**
2: Choose parameters l, n and functions f, π;
3: Choose the number t of tokens;
4: Choose the number of r indices per verification;
5: Generate challenge key k_{SRF} and master permutation key $k_{SRP;}$ using Sobol Sequence
6: **for** vector $G^{(j)}$, j←1, n **do**
7: **for** round i←1, t **do**
8: Derive $x = f_{kSRF}(i)$ and $y = f_{kSRP}(i)$ from $K_{SRP;}$
9: Compute $V_i^{(j)} = \sum^{r}_{q=1} x^q * G^{(j)}[\pi_y(q)]$;
10: **end for**
11: **end for**
12: store all the V_i s locally.
13: **end procedure**

Suppose the user wants to challenge the cloud server's t times to guaranty the Integrity of cloud data storage, he must computes the t verification tokens over each data vector $G^{(j)}$ ((j ϵ {1, …,k}) using Sobol Random Function(SRF) $f_{key}(.)$, Sobol Random Permutation(SRP) π(.), challenge key k_{SRF}, master permutation key K_{SRP}. To generate **ith** token for server j, user perform in Algorithm 1 as follows:

1) In Step 8: derive random challenge key x of $GF(2^P)$ by $x = f_{kSRF}(i)$ and master permutation key $y = f_{kSRP}(i)$ using Sobol Squence [14].
2) In step 9: Compute the set of r randomly chosen indices
{Iq ϵ [1…….l][1<q<r]} where $Iq = \pi_y(q)$.[12]
3) In step 9: Calculate tokens as
$V_i^{(j)} = \sum^{r}_{q=1} x^q * G^{(j)}[Iq]$,where $G^{(j)}[Iq] = g_{Iq}^{(j)}$.[12]

where $V_i^{(j)}$ is an element of $GF(2^P)$ with small size is the response that user expecting to receive from server j, when user challenges it on specified blocks.

Once all tokens are computed, then, we blind the each parity block $g_i^{(j)}$ in $G^{(m+1)}$,……..$G^{(n)}$ using Sobol Random Function(SRF) as follows:

$g_i^{(j)} \leftarrow g_i^{(j)} + f_{kj}(s_{ij})$, i ϵ {1…….k}[12].

where k_j is the secret key for parity data vector over $G^{(j)}$(j ϵ {m+1…..k}. This is for the protection of secret parity matrix. After, the completion of blinding parity data vectors, the user distributes encoded file across the cloud servers namely S_1, S_2……S_k to store data in cloud computing.

C. *Challenge Response Protocol*

Once data is stored in cloud, our method verifies the integrity of erasure coded data and identify the misbehaving servers in Challenge Response Protocol: the response values from servers for each challenge are not enough for determine the integrity of data in distributed storage, but also information to identify the misbehaving servers is needed.

Algorithm 2 Challenge Response Protocol

1: procedure
2: user regenerate x =fkSRF(i) and y= fKSRP(i) from KSRP using Sobol Sequence.
3: user sends {x, y} to all cloud servers ;
4:server computes $\{Ri(j)=\sum rq=1\ xq*G(j)[\pi y\ (q)]|1\leq j\leq k\}$
5: server return to Ri(j) to users;
6: for j←m+1, k do
7: $Ri(j) \leftarrow Ri(j) - \sum rq=1\ fkj(sIq,j)\cdot xq$ where Iq= πy(q);
8: end for
9: if ((Ri(1) ,……Ri(m)) · P==(Ri(m+1) ,…, Ri(k)).
10: Accept and ready for next challenge
11: else
12: for j←1, k do
13: if (Ri(j) != Vi(j)) then
14 return server j is misbehaving
15: end if
16: end for
17: end if
18: end procedure

The procedure of the ith challenge-response for cross check over the k servers in Algorithm 2 as follows:

1) In step 2: the user re-generating ith the challenge key **x**, permutation key **y** and sends to servers.
2) In step 4: Having received challenge from user, the server compute short signature over specified blocks:
$R_i^{(j)} = \sum^{r}_{q=1} x^q * G^{(j)}[\pi_y(q)]$.[12]
3) In step 7: After, receiving response $R^{(j)}$ from all servers, then user removes blind values in the response $R^{(j)}$
$R_i^{(j)} \leftarrow R_i^{(j)} - \sum^{r}_{q=1} f_{kj}(sI_q,j) \cdot x^q$ where $Iq = \pi_y(q)$;
4) In step 9: Then user check whether received response values are remain a valid codeword determined by secret parity matrix P:[12]
$(R_i^{(1)} ,\ldots\ldots R_i^{(m)}) \cdot P =^{?} (R_i^{(m+1)} ,\ldots\ldots, R_i^{(k)})$.

If the above equation holds, integrity of data storage is achieved. Otherwise it indicates that data corruption has been occurred.

Once data storage corruption has been detected in cloud successfully, then, we have to indentify the misbehaving servers using a verification token, which is achieved by verifying following equation in step 13:

$R_i^{(j)} != V_i^{(j)}$, (j ϵ {1, ……,k}).

V. PERFORMANCE EVALUATION

In this section, we analyze the performance evaluation of File Distribution and Token Pre-Computation..

A. File Distribution

We implemented a file encoding for data availability and reliability. Our experiment is conducted using java 1.6 on system with core 2 due processor running at 2.80 GHz, 4GB of RAM and 3GB of hard disk. We are considered 2 parameters for the (m+n, n) Reed-Solomon Encoding over Galois Field GF (2^8).

B. Token Pre-Computation

Like previous schemes, in our scheme also number of verification tokens t is limited decided before file distribution, our method overcome this problem by deciding number of tokens t in dynamically. For example, when t is

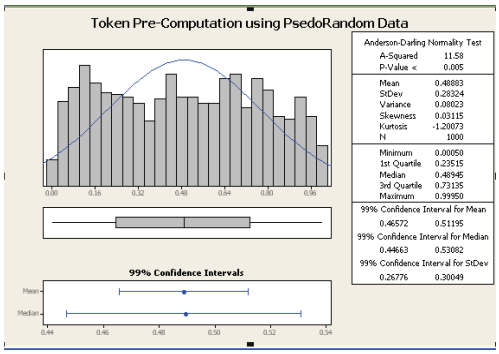

Figure 3: Token Pre-Computation using Pseudorandom Data

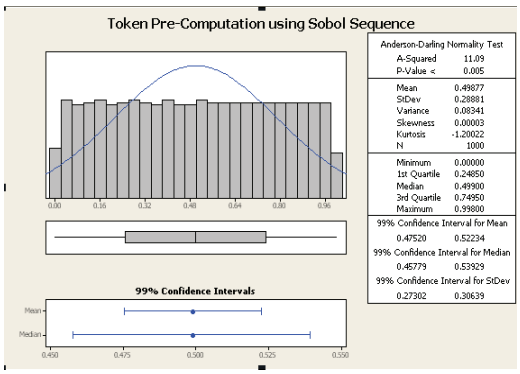

Figure 4: Token Pre-Computation using Sobol Sequence

selected to be 3650 and 5475, the data file can be verified every day for next 10 and 15 years. Practically, user select r= 460 indices, in order to identify misbehaving servers with high probability. If we use Pseudorandom Data for token Pre-Computation, it will not detect all data corruptions in cloud computing, because this method does not cover the entire data while verifying the cloud servers for data storage correctness. Compare to previous method [12], our method should detect all data corruptions in cloud computing with high probability. Figure 3 & 4 shows that, our method is **more secure** than existing method using Pseudorandom Data [12].

VI. CONCLUSION

In this paper, we proposed a more effective and flexible distributed verification scheme to address the data storage security issue in cloud computing. We rely on Reed-Solomon erasure code in file distribution to guarantee the availability and reliability of data and utilize token pre-computation using Sobol Sequence to check integrity of erasure coded data in cloud data storage. Our method achieves the availability, reliability and integrity of erasure coded data and simultaneously identifies misbehaving servers i.e. whenever data corruptions will occur during the storage correctness verification, our method should identifies the misbehaving servers. Through detailed performance analysis, we show that our scheme should provide more security to user's data in cloud computing against Byzantine failure, unauthorised data modification attacks and even server colluding attacks.

ACKNOWLEDGMENT

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression, "One of us (R.B.G.) thanks . . ." Instead, try "R.B.G. thanks". Put applicable sponsor acknowledgments here; DO NOT place them on the first page of your paper or as a footnote.

REFERENCES

[1] Cloud Computing FOR DUMMIES by Judith Hurwitz, Robin Bloor, Marcia Kaufman, and Fern Halper. WILEY INDIA EDITION.

[2] Amazon.com, "Amazon Web Services (AWS)," Online at http://aws.amazon.com, 2008.

[3] N. Gohring, "Amazon's S3 down for several hours," Online at http://www.pcworld.com/businesscenter/articl/142549/amasons_down_for_sever_hours.html, 2008.

[4] A. Juels and J. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files," *Proc. of CCS '07*, pp. 584–597, 2007.

[5] H. Shacham and B. Waters, "Compact Proofs of Retrievability," *Proc.of Asiacrypt '08*, Dec. 2008.

[6] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Cryptology ePrint Archive, Report 2008/175, 2008 http://eprint.iacr.org/.

[7] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, "Provable Data Possession at Untrusted Stores," *Proc. Of CCS '07*, pp. 598–609, 2007.

[8] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession,"*Proc. of SecureComm '08*, pp. 1–10, 2008.

[9] T. S. J. Schwarz and E. L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," *Proc.of ICDCS '06*, pp. 12–12, 2006.

[10] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard,"A Cooperative Internet Backup Scheme," *Proc. of the*

2003 USENIX Annual Technical Conference (General Track), pp. 29–41, 2003.

[11] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008, http://eprint.iacr.org/.

[12] Cong Wang,Qian wang and Kui Ren and Wenjing Lou, "Ensuring Data Storage Security in Cloud Computing , Quality of Service, 2009, IWQoS IEEE 17th International workshop ,pp 1-9,2009.

[13] J. S. Plank and Y. Ding, "Note: Correction to the 1997 Tutorial on Reed-Solomon Coding," University of Tennessee, Tech. Rep. CS-03-504, 2003.

[14] Brately P and Fox B L (1988) Algorithm 659: Implementing Sobol's Quasi-random Sequence Generator ACM Trans. Math. Software 14 (1) 88–100.

[15] L. Carter and M. Wegman, "Universal Hash unctions," *Journal of Computer and System Sciences*, vol. 18, no. 2, pp. 143–154, 1979.

[16] J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasurecoded Data," *Proc. 26th ACM ymposium on Principles of Distributed Computing*, pp. 139–146, 2007.

[17] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-Replica Provable Data Possession," *Proc. of ICDCS '08*, pp. 411–420,2008.

[18] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," cryptology ePrint Archive, Report 2006/150, 2006, http://eprint.iacr.org/.

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