Data Integrity Verification by Third Party Auditor in Remote Data Cloud

Anne Srijanya. K, N. Kasiviswanath

Abstract*—Cloud Computing gets its name as a metaphor for the internet. It is the next generation platform to provide resources as the services to the end users. In cloud storage system, the clients store their data in the server without keeping a local copy. Clients store their data in the private clouds but when storage expansion is needed they move to public clouds. Security is the major concern in the public clouds. The security mechanisms for private and public clouds are different. It may be possible that an unauthorized user can access the data from the public clouds. Hence, it is of critical importance that the client should be able to verify the integrity of the data stored in the remote un-trusted server. There may be security services offered by public clouds but they are not sufficient. In order to address the security issues, Trusted Third Party Auditing (TPA) is used as a service for private and public clouds, which offers various services to check for the integrity of the data. TPA mechanisms offer various auditing mechanisms such as read, write, update to verify the integrity of the data stored in the public clouds. We present such an auditing model based on Merkle Hash Tree. In this work we will conduct a study on possible auditing mechanisms which can be offers as a service over hybrid/public clouds. Such services can be subscribed by the users to verify the integrity of the data stored in the public clouds.

***Index Terms*— TPA, Data Storage, Public auditability, Cloud Computing, Data Dynamics.**

I. INTRODUCTION

Cloud Computing is an emerging commercial infrastructure paradigm that promises to eliminate the need for maintaining expensive computing hardware. The service provided by the cloud is very economical. The user pay only for what he used. This is a platform where data owner remotely store their data in the cloud to enjoy the high quality applications and services. The user can access the data, use the data and store the data. In a Corporate world there are large number of clients who access their data and modify their data. In Cloud, application software and services are move to the centralized large data center and management of this data and services may not be trustworthy. Through the use of virtualization and resource time-sharing, clouds address with a single set of physical resources a large user base with different needs. Thus, clouds promise to enable for their owners the benefits of an economy of scale and, at the same time, reduce the operating costs for many applications.

The increasing network bandwidth and reliable yet flexible network connections make it possible that clients can subscribe high-quality services from data and software that reside solely on remote data centers.

Although envisioned as a promising service platform for the Internet, this new data storage paradigm in "Cloud" brings about many challenging design issues which have pro-found influence on the security and performance of the overall system. One of the biggest concerns in cloud data storage is data integrity verification at untrusted servers

The integrity of data in cloud storage, however, is subject to skepticism and scrutiny, as data stored in an untrusted cloud can easily be lost or corrupted, due to hardware failures and human errors. To protect the integrity of cloud data, it is best to perform public auditing by introducing a Trusted Third Party Auditing (TPA), which offers its auditing service with more powerful computation and communication abilities. The users may resort to TPA for ensuring the storage security of their outsourced data, while hoping to keep their data private from TPA.

Security aspects like data integrity, confidentiality, and non-repudiation are seen in public cloud not much in private cloud, because the private cloud does not allow the unauthorized users to access the data but the public cloud can be accessible and seen by everyone.

Remote sensing data is acquired by the various sensors placed in the earth orbit. The data is acquired at the various ground stations and stored in the private data centers. With the increasing number of satellite sensor the data volume sizes are also increasing which needs the scalability of the compute and storage repositories at the private data centers. One of the requirements can be, to off load the data to the public cloud storages which are easily scalable. The major concern will be security aspects for the data transferred to the public storages. Hence, there is a requirement to study the security mechanisms for the public clouds with the focus to Remote Sensing data.

II. SERVICE AND DEPLOYMENT MODELS

A. Service Models

Once a cloud is established, how its cloud computing services are deployed in terms of business models can differ depending on requirements. The primary service models being deployed are commonly known as Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). In SaaS, Consumers purchase the ability to access and use an application or service that is hosted in the cloud, where necessary information for the interaction between the consumer and the service is hosted as part of the service in the cloud. In PaaS, Consumers purchase access to the platforms, enabling them to deploy their own software and applications in the cloud. The operating systems and network access are not managed by the consumer, and there might be constraints as to which applications can be deployed. In IaaS Consumers control and manage the systems in terms of the operating systems, applications, storage, and network connectivity, but they do not

Manuscript Received November, 2013.

Anne Sriyanya.K, Department of CSE ,G.Pulla Reddy Engineering College, Kurnool, Andhra Pradesh, India.

Dr.N.Kasiviswanath, Department of CSE ,G.Pulla Reddy Engineering College, Kurnool, Andhra Pradesh,India.

control the cloud infrastructure.

B. Deployment models

Deploying cloud computing can differ depending on requirements, and the following four deployment models have been identified, each with specific characteristics that support the needs of the services and users of the clouds in particular ways. In Private Cloud, The cloud infrastructure has been deployed, and is maintained and operated for a specific organization. The operation may be in-house or with a third party on the premises. In Community Cloud, The cloud infrastructure is shared among a number of organizations with similar interests and requirements. This may help limit the capital expenditure costs for its establishment as the costs are shared among the organizations. The operation may be in-house or with a third party on the premises. In Public Cloud, The cloud infrastructure is available to the public on a commercial basis by a cloud service provider. This enables a consumer to develop and deploy a service in the cloud with very little financial outlay compared to the capital expenditure requirements normally associated with other deployment options. In Hybrid Cloud, The cloud infrastructure consists of a number of clouds of any type, but the clouds have the ability through their interfaces to allow data and/or applications to be moved from one cloud to another. This can be a combination of private and public clouds that support the requirement to retain some data in an organization, and also the need to offer services in the cloud.

III. RELATED WORK

Recently much of growing interest has been pursued in the context of remotely stored data verification. Considering the role of the verifier in the model, the schemes presented earlier fall into two categories: private auditability and public auditability [1]. Schemes with private auditability can achieve higher scheme efficiency, public auditability allows anyone, not just the client (data owner), to challenge the cloud server for correctness of data storage while keeping no private information [1]. The work presented in [10] proposes a TPA mechanism that utilizes public key based homomorphic authenticator with random masking. In [11] privacy preserving TPA is proposed. The scheme proposed in [12] encryption is done at user level and then data is sent to TPA, so that data is kept secured against TPA. TPA mechanism proposed in [2] uses digital signature method for auditing the data on the cloud. The research work conducted in [7] proposes auditing mechanism that uses homomorphism token and distributed erasure coded data. The most recent work conducted in [9] proposes Trust Enhanced Third Party Auditor (TETPA) in which cloud service providers' accountability is enabled and protects cloud users' benefits.Although the existing schemes aim at providing integrity verification for different data storage systems, the problem of supporting both public auditability and data dynamics has not been fully addressed.

IV. PROBLEM STATEMENT

A. System Model

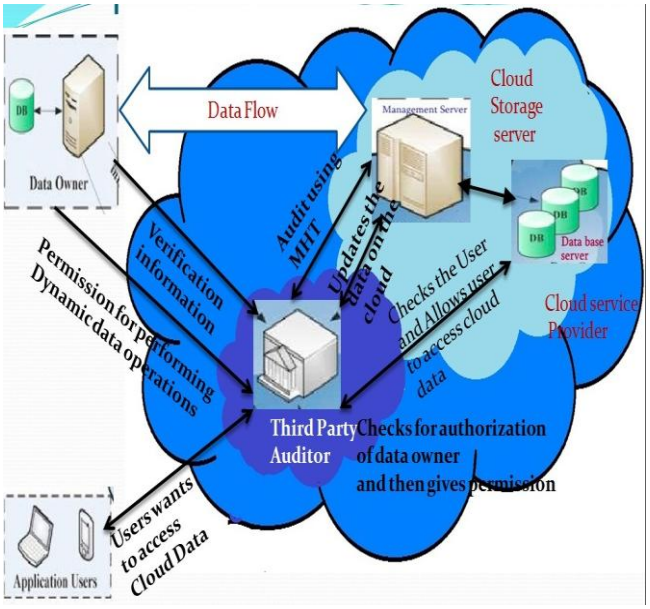


Fig 4.1: TPA which is used for auditing between public cloud and private cloud

- Client: An entity which has large data files to be stored in the cloud and relies on the cloud for data maintenance and computation can be either individual customers or organization clients are classified into 2 groups, Data Owner: Have a large amount of data to be stored in the cloud. And Application Users: Users who can access the applications with proper access permissions
- Cloud Storage server: An entity managed by CSP to provide data Storage Service.CSS is divided into two components. Management Server, manages the server and Data Server, Stores the clients data
- Cloud Service Provider: Is an Entity Which has significant storage apace and computation recourse to maintain clients data
- Third Party Auditor: Has capabilities to manage or monitor – outsourced data under the delegation of data owner. Hash values of the files are stored at TPA

B. Design Goals

Our design goals can be summarized as the following:

- Public auditability for storage correctness assurance: to allow anyone, not just the clients who originally stored the file on cloud servers, to have the capability to verify the correctness of the stored data on demand;
- Dynamic data operation support: to allow the clients to perform block-level operations on the data files while maintaining the same level of data correctness assurance. The design should be as efficient as possible so as to ensure the seamless integration of public auditability and dynamic data operation support;
- Block less verification: no challenged file blocks should be retrieved by the verifier (e.g., TPA) during verification process for efficiency concern.

V. PROPOSED SCHEME

A. Merkle Hash Tree

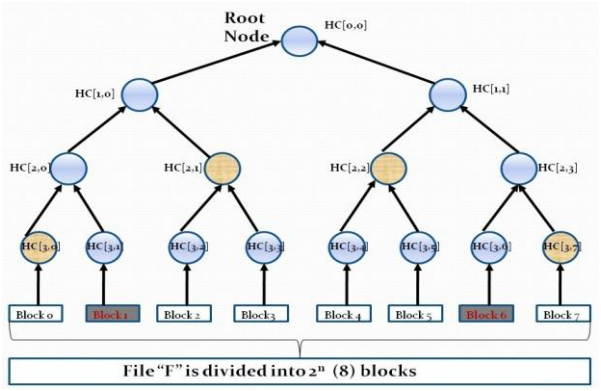


Fig 5.1 : Merkle Hash Tree

Merkle's construction built on the technique of Lamport's one-time signatures. Such a signature requires a setup stage in which many secret values are selected and the results of applying a hash function to each of them are published. To be concrete, we think of the hash function as SHA1, which outputs a 160-bit value. In its simplest form, the setup procedure for a one-time signature generates 160 secret values xi and 160 secret values yi, (we use 160 here since SHA1 outputs a 160-bit value), and the sequence of pairs HASH(xi) and HASH(yi) is made public. To sign a message m, the signer first computes the 160-bit hash or message digest of m, md = HASH(m). For each of the 160 bits of md, the signer then reveals xi if the i'th bit of md is 0, and yi otherwise. An adversary can't forge such a signature, unless he can invert the hash function HASH. However, each sequence of 320 published HASH values can only be used once. Thus, the storage associated with the original one-time signature scheme grows too large to be practical for general use

Merkle proposed a method to sign multiple messages without the enormous cost of storing two secret values per bit to be signed. His construction makes use of a binary tree, where each node is associated with a bit-string. The bit strings associated to each leaf are the hash of a secret value associated to that leaf, and each internal node of the tree is assigned HASH(L||R), where L||R represents the concatenation of the values assigned to the left and right child nodes. Rather than randomly generating and storing the secret values for each leaf, the i'th secret value can be determined from a pseudo-random generator as PRNG(secret,i). The root of the tree is made public. This key generation process is time consuming, but is a one-time cost.

A putative secret leaf value can be "authenticated" with respect to the root. Although each node value is considered to be public, the Verifier only needs to know the root value; the Prover supplies the additional public information to the Verifier: namely the values of all of the siblings of the nodes on the path to the root. The Verifier can compute the hash of the secret leaf value, then hash the result together with its sibling's value, etc., all the way up to the root. The secret value is accepted as genuine if the final value matches the published root value. The Merkle-Tree traversal problem can be thought of as the problem of easing the Prover's burden of "reminding" the verifier of the node values.

Realizing that neither storing all node values (exponential space cost in the height of the tree) nor recalculating the node values on the fly (up to exponential time cost) would be an efficient use of resources, Merkle found a way to amortize the cost of recalculating the required nodes over multiple "rounds" (1 round = output required for 1 leaf verification). For a tree of height H, Merkle's scheduling algorithm required only O(H) HASH evaluations per round, and space to store O(H^2) intermediate hash values. For medium-size trees this original algorithm already embodied a reasonable degree of storage and computation efficiency.

B. Setup

The client's public key and private key are generated by invoking KeyGen (.). By running SigGen (.), the data file F is pre-processed, and the homomorphic authenticators together with metadata are produced.

- KeyGen (1, k): The client generates a random signing key pair (spk, ssk). Choose a random α ← Zp and compute v ←g^{α}. The secret key is sk = (α, ssk) and the public key is pk = (v, spk).
- SigGen (sk, F): Given F = (m1, m2 . . . , mn), the client chooses a random element u←G. Let t be the file tag for F where t = name ‖ n ‖ u ‖ SSigssk (name‖n‖u). Then the client computes signature σi for each block mi (i = 1, 2... n) where σi ← $(H(mi)\ umi)^{\alpha}$ and denote the set of signatures by Φ = {σi}, 1 ≤ i ≤ n. The client then generates a root R based on the construction of Merkle Hash Tree (MHT), where the leave nodes of the tree are an ordered set of hashes of "file tags" H (mi) (i = 1, 2 . . . n). Next, the client signs the root R under the private key α: sigsk (H(R)) ←$(H(R))^{\alpha}$. The client sends {F, t, Φ, sigsk (H(R))} to the server and deletes {F, Φ, sigsk (H(R))} from its local storage.

C. Data Integrity Verification by TPA

The Client or TPA can verify the integrity of outsourced data by challenging the server. Before challenging, the TPA first uses spk to verify signature on t. If verification fails, reject by emitting FALSE else recover u. To generate the message "chal" the TPA (Verifier) picks a random c-element subset I= {s1, s2 ...sc} of set [1, n] where we assume s1≤.....≤sc. For each i ∈ I the TPA chooses a random element νi ← B ⊆ Zp. The message chal specifies the positions of the blocks to be checked in the Merkle hash tree. The verifier sends the chal to the {(i, νi)} s1≤i≤sc prover (server).

- GenProof (F, Φ, chal): Upon receiving the challenge chal = {(i, νi)} s1≤ i ≤ sc, the server computes, Where Both the data blocks and the corresponding signature blocks are aggregated into single block. In addition the prover will provide verifier with small amount of auxiliary information. [1]

$$\mu = \sum_{i=s_1}^{s_c} \nu_i m_i \in \mathbb{Z}_p \quad \text{and} \quad \sigma = \prod_{i=s_1}^{s_c} \sigma_i^{\nu_i} \in G,$$

- VerifyProof (pk, chal, P):Upon receiving the responses from the prover, the verifier generates root R using H (mi), Ωi} s1≤i≤sc and authenticates it by checking

$$e(sig_{sk}(H(R)), g) \stackrel{?}{=} e(H(R), g^{\alpha})$$

If the Verification fails, the Verifier rejects by emitting False. Otherwise the Verifier checks, if the output according to the equation is TRUE, otherwise FALSE.

$$e(\sigma, g) \quad \stackrel{?}{=} \quad e(\prod_{i=s_1}^{s_c} H(m_i)^{\nu_i} \cdot u^{\mu}, v).$$

VI. DATA FLOW DIAGRAMS

A data flow diagram (DFD) is a graphical representation that depicts information flow and the transforms that one applied as data moves from of a data flow diagram is also known as data flow graph or bubble.

A. Key Generation

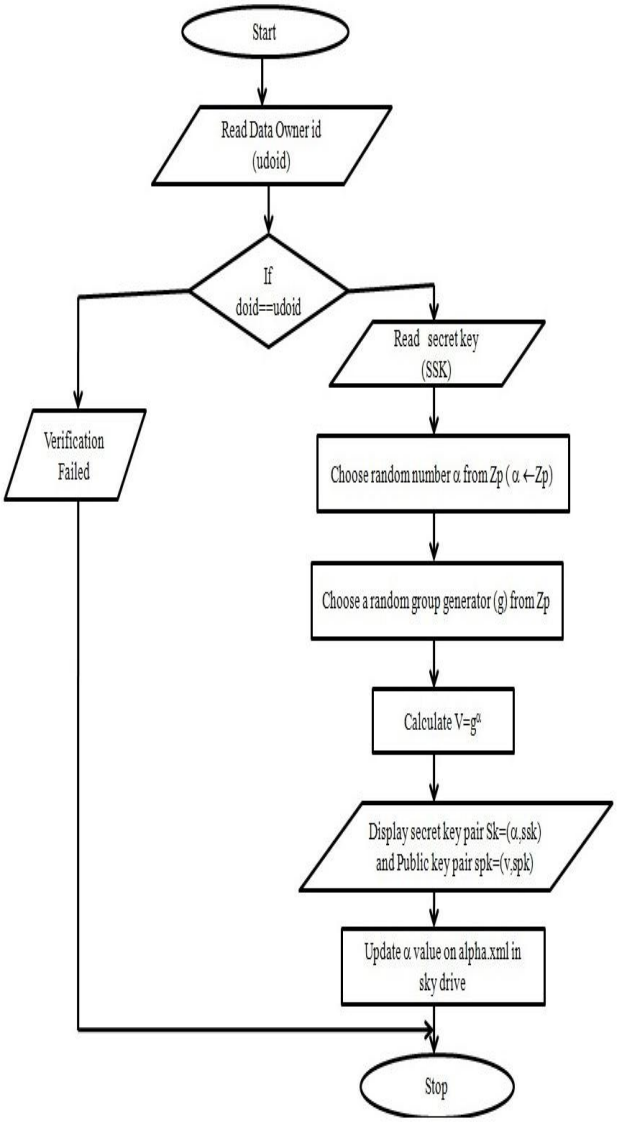


Fig6.1 : Dataflow diagram for Key Generation

B. Signature Generation

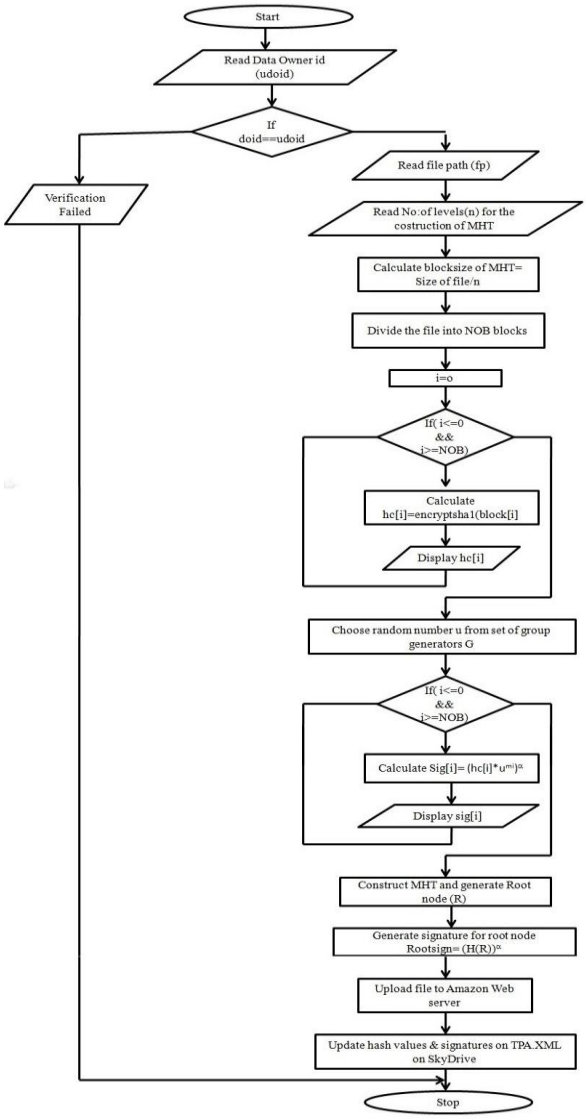


Fig 6.2 : Dataflow diagram for SignatureGeneration

C. Data Integrity Verification by TPA

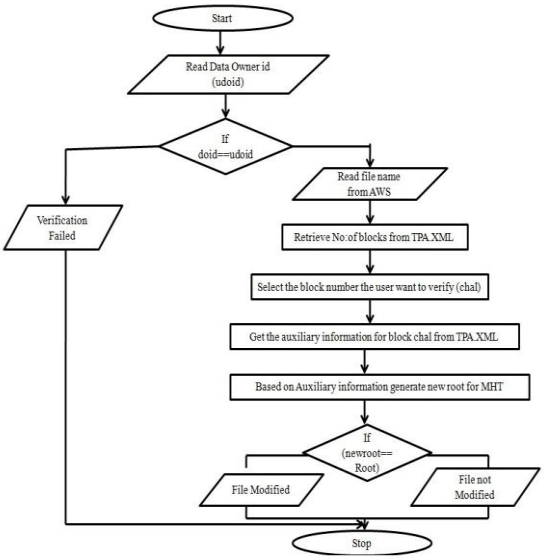
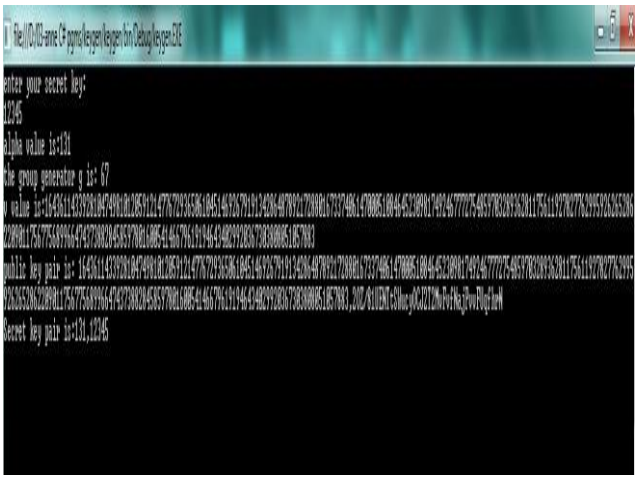


Fig6.3 : Dataflow diagram for Data Integrity Verification

VII. RESULTS

A. Key Generation



Fi 7.1 g: Generating Public key and Private key

B. Signature Generation

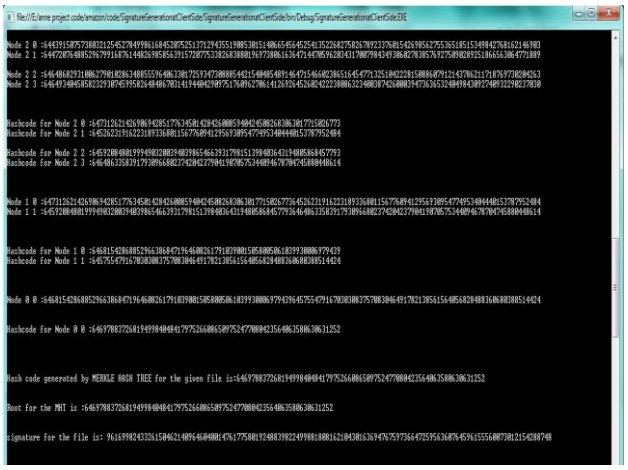


Fig7.2: Generating Signature for the given file using MHT

C. Data Integrity Verification by TPA

Fig7.3 : Data Integrity Verification by TPA(file not modified)

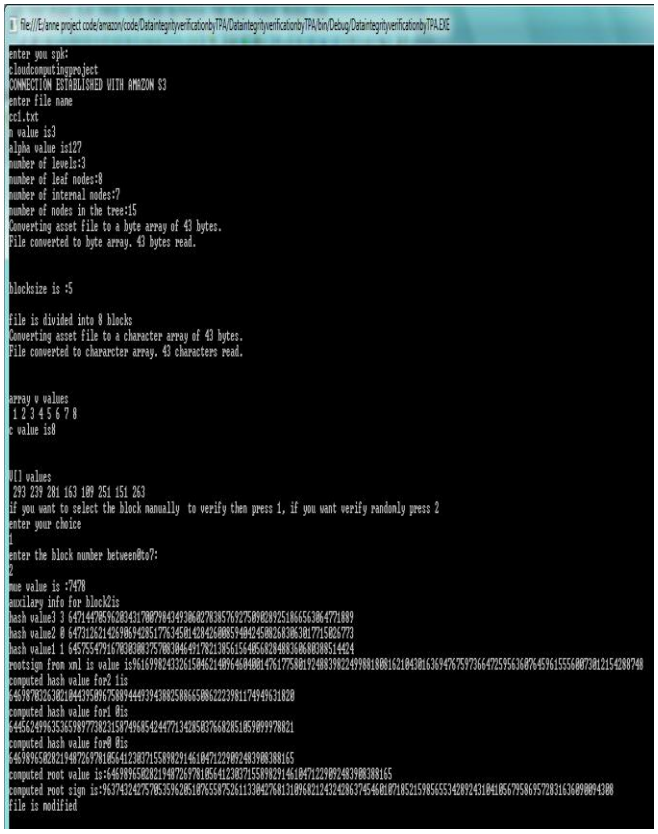


Fig 7.4: Data Integrity Verification by TPA(file modified)

VIII. CONCLUSION

In this paper we present a model for hybrid cloud in Remote sensing data using Amazon S3, OpenStack Swift and designing TPA mechanism which can be used as service and data is stored safely. Currently, the TPA and private cloud existing as service, same thing can be extended to public clouds the feasibility can be studied on the SLA's.

REFERENCES

1. Qian Wang, Cong Wang, Kui Ren, Wenjing Lou and Jin Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing"
2. K.Govinda, V.Gurunathaprasad, H.Sathishkumar, "Third Party Auditing for Secure Data Storage in Cloud through Digital Signature using RSA" International Journal of Advanced Scientific and Technical Research (Issue 2, Volume 4- August 2012) ISSN 2249-9954.
3. Wang Shao-hui, Chang Su-qin,Chen Dan-wei,Wang Zhi-we, "Public Auditing for Ensuring Cloud Data Storage Security With Zero Knowledge Privacy".
4. S. Sivachitralakshmi,T. Judgi, "A Flexible Distributed Storage Integrity AuditingMechanism in Cloud Computing", International Conference on Computing and Control Engineering (ICCCE 2012), 12 & 13 April, 2012.
5. R.Ushadevi, V. Rajamani, "A Modified Trusted Cloud Computing Architecture based on Third Party Auditor (TPA) Private Key Mechanism", International Journal of Computer Applications (0975 – 8887) Volume 58– No.22, November 2012.
6. Boyang Wang,Baochun Li,Hui Li, "Public Auditing for Shared Data with EfficientUser Revocation in the Cloud".
7. Nandeesh.B.B, Ganesh Kumar R, Jitendranath Mungara, "Secure and Dependable Cloud Services for TPA in Cloud Computing", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-1, Issue-3, August 2012.

8. Irfan Gul, Atiq ur Rehman, M Hasan Islam, "Cloud Computing Security Auditing."
9. Mortaza Mokhtari Nazarlou,Javad Badali, "A New Case for Trusted Third PartyAuditor in Cloud Computing", European Journal of Scientific Research ISSN: 1450-216X / 1450-202X Vol. 95 No 1 January, 2013, pp.152-157 © EuroJournals Publishing, Inc. 2012
10. Balakrishnan.S, Saranya.G, Shobana.S, Karthikeyan.S, "Introducing Effective Third Party Auditing (TPA) for Data Storage Security in Cloud", IJCST Vol. 2, Iss ue 2, June 2011
11. Gayatri.R , "Privacy Preserving Third Party Auditing for Dynamic Data", International Journal of Communications and Engineering Volume 01– No.1, Issue: 03 March2012.
12. Abhishek Mohta and Lalit Kumar Awasthi, "Cloud Data Security while using Third Party Auditor", International Journal of Scientific & Engineering Research,Volume 3, Issue 6, June-2012 1 ISSN 2229-5518.

AUTHOR PROFILE

Anne Srijanya.K obtained her B.Tech degree from Jawaharlal Nehru Technological University, Hyderabad in the year 2007. She is pursuing her M.Tech in Computer Science and Engineering from Jawaharlal Nehru Technological University, Anantapur, India. She presented a survey paper at a national level conference.

Dr.N.Kasiviswanath received hisB.E. degree in Computer Science& Engineering from Marthwada University, M.S. degree in ComputerScience & Engineering from BITS,Pilani and the Ph.D. degree from Rayalaseema University, Kurnool. Hehas 20 years of teaching experience.He has published 25 research papers in National/International journals/conferences. At present, he is working as Professor and Head of Computer Science & Engineering Department, G. Pulla Reddy Engineering College, Kurnool.