ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

Remote Sensing Data as a Service in Hybrid Clouds: Security Challenges and Trusted Third Party Auditing Mechanisms

Ms.G. Divya Zion[1], **Dr. D.Kavitha**[2]

[1]M.Tech (CSE) Student, G. Pulla Reddy Engineering College, KURNOOL, Andhra Pradesh. India.

[2]Professor, Dept. of CSE, G. Pulla Reddy Engineering College, KURNOOL, Andhra Pradesh. India.

Abstract – **Cloud Computing is the next generation platform to provide resources as the services to the end users. In this paper we present a service for Trusted Third party Auditing (TPA) mechanisms. Hybrid cloud connects the private and public clouds, and it is useful when the resource expansion is needed at the private cloud but if there are limitations, then the public clouds resources can be utilized. Here, we present such a model where the data is stored in the cloud storages, in the private cloud. In case of storage expansion, it is proposed to use the public cloud storages. Security is the major concern in the public clouds. The security issues for the data can be addressed using Auditing mechanisms. Here, we present such auditing model based on Merkle Hash Tree. Auditing mechanisms presents the results like data transfer, retrieval and searching mechanism without fully decrypting the data, as well to retain the data integrity, non-repudiation of the data.**

***Keywords*: TPA, Merkle Hash Tree, Data Storage, Cloud Computing**

I. INTRODUCTION

Cloud computing is the next generation IT technology to provide the compute, storage and network resources as a utility over the public network such as Internet. Public cloud is hosted, operated and managed by a third party vendor from one or more data centers. The services are offered to multiple customers in this public cloud security management and day-to-day operations are related to the third party vendor, who is responsible for the public cloud service offering. In public cloud the resources are dynamically provisioned on a fine grained, self service basis over the internet via web applications or web services. You can choose from various versions of Public Cloud offerings. They can be delivered in the form of hosting, Software-as-a-Service, or even storage. Hosting provides entities with the ability to create and manage their virtual servers to run their desired applications. Software-as-a-Service, or SaaS, is a web based application that individuals can sign-up to use over the Internet, however all management and information is controlled and maintained by the service provider. Private clouds has the storage infrastructure associated is dedicated to a single organization and is not shared with any organizations. Private cloud is used to describe offering that emulate cloud computing on private network. The main reason for opting for Private Cloud services is to provide a more controlled and custom environment where performance, security, and compliance are mandatory. Resources are dedicated or reserved for only a single customer to use which allows control of performance to meet the needs of more sophisticated and advanced applications. Hybrid cloud consists of multiple internal and external providers from a particular organization. In Hybrid cloud, organizations might run non-core applications in public cloud, while maintaining core applications and sensitive data in-house a private cloud.

Since Mainly security aspects like data integrity, confidentiality, and non-repudiation are seen in public cloud not much in private cloud, since the private cloud does not allow the unauthorized users to access the data but the public cloud can be accessible and seen by everyone. In this paper we present the security mechanism in such a way that encrypting the data and placing into the public cloud which is the general method and we find this method even unsafe so we do the process through Merkle hash tree, where the data is divided into blocks and then we do hashing on each of the block and the place the data in the public cloud this process allows us to find even when the data is undamaged and unaltered using Merkle Hash Tree [1]. When offloading this data to Amazon S3 [23] which is the public cloud we use homomorphic authenticators [21] where data is stored

securely and the auditing mechanisms are seen.
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. CLOUD ARCITECTURE OVERVIEW

The cloud architecture has three layers IaaS (Infrastructure as a service), SaaS (software as a service), PaaS (platform as a service). IaaS is determined as utility computing model and it is a cloud computing service model in which hardware is virtualized in the cloud and the service vendor owns the equipment like servers, storage, and network infrastructure. The virtualized resources are mapped to real system, when the client interact with an IaaS service and requests resources from the virtual systems, those requests are redirected to the real servers that do the work, the IaaS has public clouds which are Amazon(EC2, S3, SQS, cloud front), proof point, right scale [20].

PaaS provides the development tools for application/service design on time without installation the development can be carried out. For eg: provide .Net, SQL as online tools no need of installation.

In SaaS consumer uses an application, but does not control the hardware, operating system or network infrastructure on which it is running. A SaaS deployment does not require any hardware and can run over the existing internet access infrastructure. The PaaS providers are Microsoft Windows Azure, Google, Sales force.com, and SaaS providers are Google and workday.

III. RELATED WORK

Recently, much of growing interest has been pursued in the context of remotely stored data verification [5]–[13], [14][15]–[18]. Ateniese et al. [8] are the first to consider public auditability in their defined "provable data possession" (PDP) model for ensuring possession of files on untrusted storages. In their scheme, they utilize RSA-based homomorphic tags for auditing outsourced data, thus public auditability is achieved. However, Ateniese et al. do not consider the case of dynamic data storage, and the direct extension of their scheme from static data storage to dynamic case may suffer design and security problems. In their subsequent work [14], 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 does not support fully dynamic data operations, i.e., it only allows very basic block operations with limited functionality, and block insertions cannot be supported. In [16], Wang et al. consider dynamic data storage in a distributed scenario, and the proposed challenge-response protocol can both determine the data correctness and locate possible errors. Similar to [14], they only consider partial support for dynamic data operation. Juels et al. [9] describe a "proof of retrievability" (PoR) model, where spot-checking and error correcting codes are used to ensure both "possession" and "retrievability" of data files on archive service systems. Specifically, some special blocks called "sentinels" are randomly embedded into the data files F for detection purpose, and F is further encrypted to protect the positions of these special blocks. However, like [14], the number of queries a client can perform is also a fixed priori, and the introduction of pre-computed "sentinels" prevents the development of realizing dynamic data updates. In addition, public auditability is not supported in their scheme. Shacham et al. [10] design an improved PoR scheme with full proofs of security in the security model defined in [9]. They use publicly verifiable homomorphic authenticators built from BLS signatures [13], based on which the proofs can be aggregated into a small authenticator value, and public retrievability is achieved. Still, the authors only consider static data files. . Erway et al. [12] was the first to explore constructions for dynamic provable data possession.

They extend the PDP model in [8] 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. To support updates, especially for block insertion, they eliminate the index information in the "tag" computation in Ateniese's PDP model [8] and employ authenticated skip list data structure to authenticate the tag information of challenged or updated blocks first before the verification procedure. However, the efficiency of their scheme remains unclear. 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. How to achieve a secure and efficient design to seamlessly integrate these two important components for data storage service remains an open challenging task in Cloud Computing.

Q.Wang [15] proposed **two** basic solutions (i.e., the MAC-based and signature- based schemes) for realizing data auditability and discuss their demerits in supporting public auditability and data dynamics. Secondly, generalize the support of data dynamics to both proof of retrievability (PoR) and provable data possession (PDP) models and discuss the impact of dynamic data operations on the overall system efficiency both. In particular, they emphasize that while dynamic data updates can be performed efficiently in PDP models more efficient protocols need to be designed for the update of the encoded files in PoR models.

The public cloud which we work in this paper is Amazon S3 (Simple Storage Service) which uses REST API's. The S3 has simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits to developers [23]. The private cloud is OpenStack Swift, Swift is a highly available, distributed, eventually consistent object/blob store[22]. Organizations can use Swift to store lots of data efficiently, safely, and cheaply. OpenStack Swift is also known as OpenStack Storage, in addition to traditional enterprise-class storage technology, many organizations now have a variety of storage needs with varying performance and price requirements. OpenStack has support for both Object Storage and Block Storage, with many deployment options for each depending on the usecase. Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire [22].

IV. PROBLEM STATEMENT

To provide the remote sensing data as a service to the users enables to retrieve the data of interest and carry out the analysis process. Here, the data is stored in the cloud storage repositories which can be accessed by HTTP mechanisms such as SOAP, REST. When the data is available in the private cloud storages security is maintained but, in the scenario uploaded to the public clouds security becomes a major concern. The major security concerns are 1) Data integrity 2) Confidentiality 3) Non-repudiation. The major problem which we present in this paper is that as the storage limitation is less in the private clouds so we offload the data to the public cloud, there should be an interface between the public cloud and the private cloud for retrieving the data and Third Party Auditor audits the data whether the data which is present in the public cloud is affected or not. The offloading of data is discussed in three ways firstly, the data file is divided into blocks and then uploading the data directly to the public cloud and then computing the signatures based on the file index information "i". Therefore, once a file block is inserted, the computation overhead is unacceptable since the signatures of all the following file blocks should be re-computed with the new indexes. This process takes lot of time to compute the signatures for all the data blocks of files and we cannot even find even if any block is modified.

Secondly, assume the outsourced data file F consists of a finite ordered set of blocks m1, m2… mn. To ensure the data integrity is to pre-compute MACs for the entire data file. Specifically, before data outsourcing, the data owner pre-computes MACs of F with a set of secret keys and stores them locally. During the auditing process, the data owner each time reveals a secret key to the cloud server and asks for a fresh keyed MAC for verification. This approach provides deterministic data integrity assurance as the verification covers all the data blocks. However, the number of verifications allowed to be performed in this solution is limited by the number of secret keys. Once the keys are exhausted, the data owner has to retrieve the entire file of F from the server in order to compute new MACs, which is usually impractical due to the huge communication overhead. Moreover, public auditability is not supported as the private keys are required for verification.

Thirdly, we encrypt the entire file and offload the data but the data can be decrypted and seen easily and modified by the attackers this doesn't support public auditability.

So we propose a new solution in this paper is to compute signatures instead of MACs to obtain public auditability. The data owner precomputes the signature of each block mi ($i \in [1, n]$) and sends both F and the signatures to the cloud server for storage. To verify the correctness of F, the data owner can adopt a spot-checking approach, i.e., requesting a number of randomly selected blocks and their corresponding signatures to be returned. This basic solution can provide probabilistic assurance of the data correctness and support public auditability.

V. SYSTEM DESIGN

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

We propose the scheme where the file F is divided into blocks and compute signatures for each block and sends both the signatures and the blocks F to the cloud server for storage this is done using Merkle hash tree. So the cloud server discussed in this paper is Amazon S3 and the private cloud is OpenStack Swift where a TPA is placed

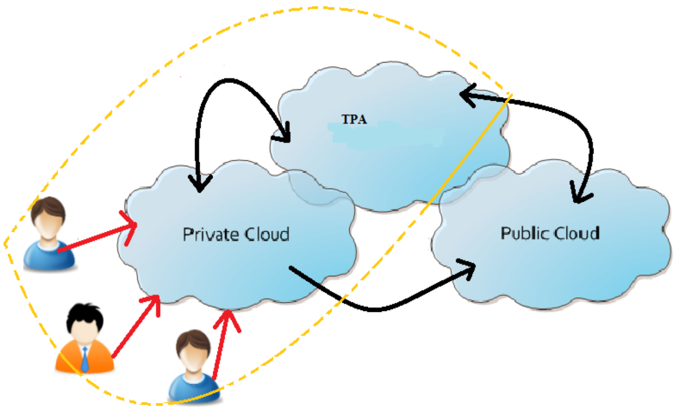

Fig1. TPA which is used for auditing between public cloud and private cloud

The Figure1 explains that the users who have their File in the private cloud sends the File which is divided into blocks to the public cloud and the TPA.whn the user wants to verify the blocks are modified or not the TPA plays a major role, the TPA which has a local copy i.e; the copy sent by the private cloud is stored and it asks the public cloud the signatures of the blocks of file and verifies the it compares the results with local copy which it has and then it sends to the private cloud and to users.

A. Merkle Hash Tree

A Merkle Hash Tree (MHT) [17] is intended to efficiently and securely prove that a set of elements are undamaged and unaltered. It is constructed as a binary tree where the leaves in the MHT are the hashes of authentic data values. The verifier with the authentic hr requests for {x2, x7} and requires the authentication of the received blocks [1]. MHT is commonly used to authenticate the values of data blocks. However, in this paper we further employ MHT to authenticate both the values and the positions of data blocks. We treat the leaf nodes as the left-to-right sequence, so any leaf node can be uniquely determined by following this sequence and the way of computing the root in MHT.

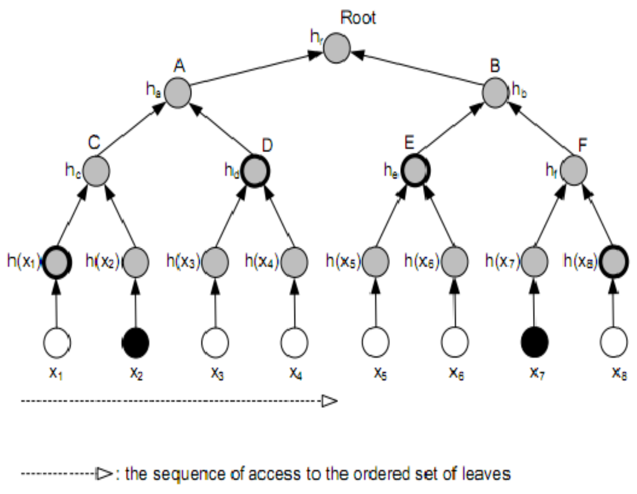

Fig2. Merkle Hash Tree Authentication of data elements

In Figure2 the leaf nodes h(x1)......h (xn) as the left-to-right sequence, we assume that file F (potentially encoded using Reed-Solomon codes [18]) is divided into n blocks m1, m2, mn, where mi ∈ Zp and p is a large prime. Let g be the generator of G. "h"is a cryptographic hash function.

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 (1k): The client generates a random signing key pair (spk, ssk). Choose a random α ← Zp and compute v $\leftarrow g^{\alpha}$. 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)) $\leftarrow (H(R))^{\alpha}$. The client sends {F, t, Φ, sigsk (H(R))} to the server and deletes {F, Φ, sigsk (H(R))} from its local storage.

C. Uploading to AMAZON S3

After performing the Setup phase the File F (m1, m2… mn) has to be uploaded to public cloud Amazon S3 from the local file by giving the path of the file. We upload the entire file by dividing the text file into blocks and the hash of each blocks are uploaded and stored. Now when the user wants to download or retrieve the data, the TPA comes into picture which we call it as auditing.

D. Default Integrity Verification

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,

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

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]

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

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

If the Verification fails, the Verifier rejects by emitting False. Otherwise the Verifier checks,

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

If the output according to the equation is TRUE, otherwise FALSE.

E. Data Operations with Integrity Assurance

The data operations including data modification, data insertion, and data deletion for cloud data storage which are the major security concerns can be efficiently handled and when the data is modified then it can be known.

VI. 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. 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] R. C. Merkle, "Protocols for public key cryptosystems," Proc. Of IEEE Symposium on Security and Privacy'80, pp. 122–133, 1980.
[2] M.Bellare and P.Rogaway,"Random oracles are practical: A paradigm for designing efficient protocols," in Proc of CCS 93, 1993
[3] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in Proc. of ASIACRYPT'01. London, UK: Springer-Verlag, 2001, pp. 514–532.
[4]S.Lin and D.J.Costello, Error Control Coding, Second Edition.Upple Saddle River, NJ, USA: Prentice Hall,Inc, 2004.
[5] M. Naor and G. N. Rothblum, "The complexity of online memory checking," in Proc. of FOCS'05, Pittsburgh, PA, USA, 2005, pp.573–584.
[6] A. Oprea, M. K. Reiter, and K. Yang, "Space-efficient block storage integrity," in Proc. of NDSS'05, San Diego, CA, USA, 2005.
[7] T. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in Proc. of ICDCS'06, Lisboa, Portugal, 2006, pp. 12–12.
[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. of CCS'07. New York, NY, USA: ACM, 2007, pp. 598–609.
[9] A. Juels and B. S. Kaliski, Jr., "Pors: proofs of retrievability for large files," in Proc. of CCS'07. New York, NY, USA: ACM, 2007,pp. 584–597.
[10] H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. of ASIACRYPT'08. Melbourne, Australia: Springer-Verlag, 2008, pp. 90–107.
[11] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: Theory and implementation," Cryptology ePrint Archive, Report 2008/175, 2008.
[12] E.-C. Chang and J. Xu, "Remote integrity check with dishonest storage server," in Proc. of ESORICS'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 223–237.
[13] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008.
[14] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. of SecureComm'08.New York, NY, USA: ACM, 2008, pp. 1–10.
[15] 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. of ESORICS'09. Saint Malo, France: Springer- Verlag, 2009, pp. 355–370.
[16] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in Proc. of IWQoS'09, Charleston, South Carolina, USA, 2009.
[17] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. of CCS'09. Chicago, IL, USA: ACM, 2009.
[18] K. D. Bowers, A. Juels, and A. Oprea, "Hail: A high-availability and integrity layer for cloud storage," in Proc. of CCS'09. Chicago, IL, USA: ACM, 2009, pp. 187–198.
[19] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and secure sensor data storage with dynamic integrity assurance," in Proc. Of IEEE INFOCOM'09, Rio de Janeiro, Brazil, April 2009, pp. 954–962.
[20]Trusted Data sharing over untrusted cloud storage providers 2010 IEE Gansen Zhao, Chumming rong, Jin Li

[21] Homomorphic Authentication with Random Masking Technique ensuring privacy and security in Cloud computing,2012 jachak K.B,Korde, Gorphade P.P and Gagare G.J
[22] High Performance Private Cloud for satellite Data Processing-Engineering in cloud, 2012 by Raghavendra Kune, Ankit Chaudri, Pramod Kumar Konugurthi, Geetha Vardhan, at Advanced Data processing Research Institute, Dept.Of Space.
[23] http://aws.amazon.com/documentation/

Biography

Miss. G. Divya Zion has completed B. Tech in Computer Science and Engineering from St. Johns College Of Engineering and Technology, Yemmiganur, Kurnool in the year 2010, affiliated to JNTUA. Presently she is pursuing her M. Tech from G. Pulla Reddy Engineering College (Autonomous), affiliated to JNTUA.

Dr.D. Kavitha obtained her B.Tech degree from Sri Krishna Devaraya University, Anantapur and M.Tech degree from Jawaharlal Nehru Technological University, Anantapur in the year 2001 and 2005 respectively. She obtained Ph.D in Computer Science from Sri Krishna Devaraya University, Anantapur in 2012. She is presently working as Professor in the Department of Computer Science and Engineering at G. Pulla Reddy Engineering College, Kurnool, and Andhra Pradesh, India. She has presented nine research papers in various national and international journals so far her research areas include Computer Networks and Network Security.