**Enabling Ternary Hash Tree based Integrity Verification for Secure Cloud Data Storage**

**Alternative Title: Creating Secure Clouds by Continuous Auditing Using Ternary Hash Tree**

**Aim**

 The main aim of this project is to provide a reliable and secure cloud service and also increase trustworthiness of certifications by Continuous Automatic Auditing.

**Introduction**

Cloud Computing enables the remote users to access data, services, and applications in on-demand from the shared pool of configurable computing resources, without the consideration of storage, hardware and software management. On the other hand, it is not easy for cloud users to identify whether Cloud Service Provider's (CSP) tag along with the data security legal expectations. So, cloud users could not rely on CSP's in terms of trust. So, it is significant to build a secure and efficient data auditing framework for increasing and maintaining cloud users trust with CSP. Researchers suggested introducing Third Party Auditor (TPA) on behalf of cloud user for verifying the outsourced data integrity, which may reduce the computation overhead of cloud users. In this work, we proposed a novel integrity verification framework for securing cloud storage based on Ternary Hash Tree (THT) and Replica based Ternary Hash Tree (R-THT), which will be used by TPA to perform data auditing. Differing from existing work, the proposed framework performs Block-level, File-level and Replica-level auditing with tree block ordering, storage block ordering for verifying the data integrity and ensuring data availability in the cloud. We further extend our framework to support error localization with data correctness, dynamic updates with block update, insert and delete operations in the cloud. The structure of THT and R-THT will reduce the computation cost and provide efficiency in data updates compared to the existing schemes. The security analysis of the proposed public auditing framework indicates the achievement of desired properties and performance has been evaluated with the detailed experiment set. The results show that the proposed secure cloud auditing framework is highly secure and efficient in storage, communication and computation costs.

**Existing System**

In cloud computing, remote data integrity checking is an important security problem. The client’s massive data is outside his control. The malicious cloud server may corrupt the client’s data in order to gain more benefits. However, cloud services are part of an ever changing environment, resulting from fast technology life cycles and inherent cloud computing (CC) characteristics, like on-demand provisioning and entangled supply chains. Hence, such long validity periods may put in doubt reliability of issued certifications. And also cloud service customers do not longer possess their data locally, assuring that their data is being correctly stored and integrity is maintained in cloud environments is of critical importance. Data integrity may be threatened by, for example, malicious insiders, data loss, technical failures, and by external attackers.

**Problem Definition**

1. Long validity periods may put in doubt reliability of issued certifications.
2. Data Integrity.
3. Procrastinating third-party auditors.

**Proposed System**

In MultiCloud environment, remote data integrity checking is required to secure user’s data. User will upload file to Cloud. This file is split into blocks using Dynamic Block generation Algorithm. The Blocks are stored in Ternary Hash Tree (THT) format. The blocks have a parents node and child node. File Allocation Table (FAT) File System has proper Indexing and Metadata’s for the different Chunks of the Cloud Storage. Here the auditor agrees to inspect logs, which are routinely created during monitoring operations by services providers to assess certification adherence. If Attacker corrupts data in MultiCloud, the continuous auditing process helps the verifier to perform Block and File level checking for remote data Integrity Checking using Verifiable Data Integrity Checking Algorithm. The auditing processes have a flow, first the parent block checking. If the parent block have any corrupted file then the child node auditing. If the child nodes have any corrupted file the File recovery is done by the Verifier automatically if the data gets corrupted during checking. Users can complaint cloud for file recovery.

**Advantages:**

* The proposed framework performs Block-level, File-level and Replica-level auditing with tree block ordering.
* Our framework to support error localization with data correctness, dynamic updates with block update, insert and delete operations in the cloud.

**Modules**

* Server Configuration
* Data Upload and Block Split
* Data Integrity Checking and Update
* File Recovery

**Server Configuration**

 Admin configure Multi-Cloud server setup. Server IP Address and Port number is given by the admin for each Cloud. Now a Server Architecture is created for Multi-Cloud Storage. If the admin has to reconfigure the old Multi-Cloud server setup, it can be done. For old server setup, FAT file can be modified or remain same. Audit time will be set by the admin for Data Integrity checking process.

**Data Upload and Block Split**

 User has an initial level Registration Process at the web end. The users provide their own personal information for this process. The server in turn stores the information in its database. After Registration, user can upload files to the server. Uploaded files will be stored in a Server. When the user upload the data to different cloud by the time it is splitted into different blocks using dynamic block generation Algorithm and each block will be appended with Signatures before storing the data in FATFS. Signature generated using MD5 Algorithm. Also the data gets encoded using for Base64 Algorithm.

**Data Integrity Checking and Update**

FATFS has proper Indexing and Metadata’s for the different Chunks of the Data that is being uploaded by User. Verifier performs Remote Integrity Checking on Cloud Data. Cloud allocates random combination of all the blocks to the Verifier, instead of the whole file is retrieved during integrity checking. This is to protect user privacy from a third party (Verifier). Verifiable Data Integrity Checking Algorithm is done in two steps: Block Checking and File Checking. In Block Checking step: Three signatures are generated for Block level Checking.

* A signature of a block retrieved from a FATFS
* A new signature is generated for block to be checked
* A Signature is retrieved from the block appended with the signature which is stored in the Cloud

 The above three signatures are cross checked for Block level Integrity Checking. And the block contents are appended to verify with File level Integrity Checking.

**File Recovery**

Attacker can corrupt data in any one of the cloud servers. On Data Integrity Checking done by the Verifier, Verifier informs Corrupted blocks to the Cloud. Recovery Process will be done by the verifier automatically when data gets corrupted. User can complaint to the Cloud if the user file get corrupted (Verifier doesn’t perform checking on this file).Whenever user access file, Blocks will be reallocated dynamically to provide access confidentiality in cloud and FAT File System will get updated.Auditor will monitor the cloud continuously and they provide the certificate based on the cloud performance. When new user join in the cloud they will read the certificate and then they can create an account in the cloud.

**Enhancement**

* File Recovery
* Access Confidentiality: Dynamic reallocation of blocks at every access.
* Privacy preservation on Public Auditing
* Dynamic Block Split

**Algorithms**

* Base64 Algorithm
* Message Digest(MD5)
* Dynamic block generation

**Software Requirements**

* Windows XP and Above
* JDK 1.7
* Tomcat 6.0
* Tomcat 7.0
* MySql 5.0

**Hardware Requirements**

* Hard Disk : 200GB and Above
* RAM : 2GB and Above
* Processor : i3 and Above

**Technology Used**

* J2EE (JSP, Servlet)
* Struts Framework
* JavaScript , Ajax , HTML ,CSS, jquery
* Web Services (JAX -WS), JSON.

### Conclusion

The data auditing in file level, block level, and replica level are achieved for verifying data integrity of the entire file. Few blocks for frequent audit tasks making it computationally efficient and replica level auditing to ensure data consistency across all the replicas in the cloud respectively. Further, the corrupted blocks identified during auditing is localized and corrected to suit the need for real-time applications. Moreover, public auditing preserves the privacy of user data from TPA through the random ordering of the blocks being unknown to TPA and CS. Further data dynamics is performed maintaining public verifiability on the same with reduced complexity which was better than the existing schemes.

### Future Enhancement

The future work is to extend the framework for data sharing where data integrity is verified over the shared data across the user group.

**System Architecture**

 User

 Verifier

Attacker

Corrupt

Data integr ity checking

Multi-server

Block Split using Ternary Hash Tree

Signature generation

Encoding

S1

S2

Sn

Block distribution

Web Server

Access file

Dynamic reallocation of blocks

 FAT

Blocks and

Signature

signature

Is Data Corrupt

Yes

No

Recover Data

 Upload file

Create Account