A Collaborative Auditing Blockchain For Trustworthy Data Integrity And Recovery In The Cloud Computing System

S.Sriram[1], Mr.S.J.Vivekanandan[2]
[1]Dept of Computer Science and Engineering
[2]Assistant Professor, Dept of Computer Science and Engineering
[1, 2] Chennai, Tamilnadu,India

Abstract- *In this paper, we propose a privacy-preserving mechanism that supports public auditing on shared data stored in the cloud. The existing system doesn't support multiple auditing tasking which makes it difficult for auditors to verify shared data. All the user data gets leaked to the auditor due to which there is data integrity and privacy issues. In order to audit shared data integrity without retrieving the entire data and distinguish the signer, we construct homomorphism authenticators using MD5 and SHA-256. With our mechanism, the identity of the signer on each block in shared data is kept private from public verifiers, who are able to efficiently verify shared data integrity without retrieving the entire file. In addition, our mechanism is able to perform multiple auditing tasks simultaneously instead of verifying them one by one to support batch auditing. In our future work, we will continue to study impact of traceability, which means the ability for the group manager to reveal the identity of the signer based on verification of metadata in some special situations.*

Keywords- Advanced Encrypted Standard, Cloud comput- ing,Data Integrity,Message-Digest 5 Algorithm

I. INTRODUCTION

CLOUD service provider offers data storage services in an efficient and scalable way with a much lower marginal cost than traditional approaches. The shared file is divided into a number of small blocks, where each block is independently signed by two users with existing public auditing solutions. Once a block is modified by a user, the user needs to sign the modified block using his/her private key. Eventually, different blocks are signed by different users due to the modification introduced by these two different users. Hence, to correctly audit the integrity of the entire data, a public verifier needs to choose the appropriate public key for each block (e.g., a block signed by Alice can only be correctly verified by Alice's public key). Then the unique binding between an identity and a public key via digital certificates under public key infrastructure (PKI) helps the public verifier to identify the signer on each block.

In this paper, we mainly focus on solving the privacy issue on shared data using privacy-preserving public auditing mechanism. The public verifier is able to verify the integrity of shared data without retrieving the entire data, also by keeping the identity of the signer on each block in shared data, private from the public verifier.

II. EXISTING SYSTEM

The existing mechanism has a significant privacy issue. In the case of shared data, the information of the users is leaked to public verifiers. As users no longer physically possess the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted.

The traditional approach for checking data correctness is to retrieve the entire data from the cloud, and then verify data integrity by checking the correctness of signatures. To securely introduce an effective third-party auditor (TPA), the following two fundamental requirements have to be met:

- TPA should be able to efficiently audit the cloud data storage without demanding the local copy of data, and introduce no additional on-line burden to the cloud user
- The third-party auditing process should bring in no new vulner- abilities towards user data privacy

III. PROPOSED SYSTEM

The proposed system uses a privacy-preserving public au- diting mechanism for shared data in the cloud. In order to audit shared data integrity without retrieving the entire data and distinguish the signer, we construct homomorphism authenticators using Message- Digest algorithm 5 (MD5) and SHA-256. We further extend our

mechanism to support batch auditing to improve the efficiency of verifying multiple auditing tasks. In our system admin generates a key and it reaches the users with SMTP (Simple Mail Transfer Protocol). Basically, SMTP are a set of commands that certify and direct the transfer of electronic message. Once configured the settings for your e-mail program, you always ought to set the SMTP server to your native net Service Provider's SMTP settings. However, the incoming mail server (POP3 or IMAP) ought to be set to your mail account's server, which can differ based on the SMTP server.

IV. REQUIREMENT SPECIFICATION

A. Hardware Requirements

- 4GB RAM
- 80GB Hard Disk
- Above 2GHZ Processor

B. Software Requirements

- Language – Java (JDK 1.7)
- OS – Windows 7 32bit
- Database – MySQL Server
- IDE – NetBeans IDE 7.1.2

C. Technologies

JAVA - The Java programming language is a high- level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance

In Java programming language, all source codes are developed in Eclipse IDE. Those source files are then compiled into .class files by the javac compiler A .class file does not contain code that is native to your processor; it instead contains byte codes that is executed by Java Virtual Machine (JVM). The java launcher tool then runs your application with an instance of the JVM.

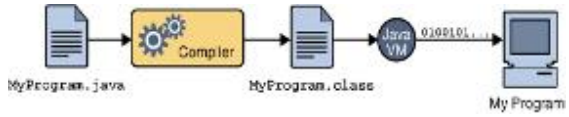

Fig. 1: An overview of the software development process

Java Server Pages (JSP) - Java Server Pages (JSP) is a Java technology that are used by software developers to dynamically generate XML, HTML or other types of documents in response to a Web client request. The technology allows Java code and certain pre-defined actions to be embedded into static content.

The JSP syntax adds additional XML-like tags, called JSP actions, that invokes built-in functionality. In addition, the technology allows for the creation of JSP tag libraries that act as extensions to the standard XML or HTML tags. Tag libraries provide platform independent way to extend the capabilities of a Web server.

JSPs are compiled into Java Servlet by a JSP compiler to generate a servlet in Java code which is compiled by the Java compiler, or it may generate byte code for the servlet directly. JSPs can also be interpreted on-the-fly that reduces the time taken to reload changes. Java Server Pages (JSP) technology provides a simplified and fast way to create dynamic web content. JSP enables rapid development of web-based applications that are server and platform-independent

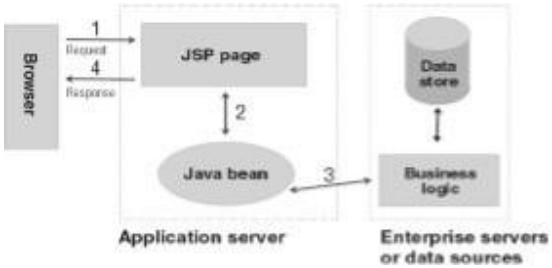

Fig. 2: Architecture of JSP

database, regardless of what database management software is used to control the database. In this way, JDBC is platform independent

V. SYSTEM OVERVIEW

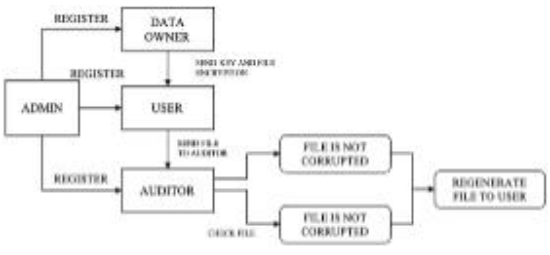

Fig. 3: System Architecture

In this system, admin has complete access to register and generate IDs and file keys for Data owners, users and auditors. When the user wants the specific file, only the user's corresponding owner owns its right to upload the

corresponding file and sends the key and encrypted file to the user, which the user will send it to auditor to verify the file for security. Once the auditor checks the file, the auditor sends back the file to the user in case of file is not corrupted. Else, the auditor will retrieve the backup file and regenerate it and send back to the user

A. Data Flow Diagram

Data flow diagrams (DFD) illustrate how data is handled by a system in terms of inputs and outputs. Data flow diagrams is used to provide a clear representation of any business function. The technique starts with an overall picture and continues by analyzing each of the functional areas of interest. This analysis can be carried out in the level of detail required. The technique makes use of a method called top-down expansion to conduct the analysis. DFD is also known as a Process Model.

Servlets – Front End - A Servlet is an object that receives a request and generates a response. The basic Servlet package defines Java objects based on Servlet requests and responses, as well as objects to reflect the Servlet configuration parameters and execution environment. Servlet may be packaged in a WAR file as a Web application. Often Servlet are used in addition with JSPs in a pattern called ”Model 2”, which follows the model-view-controller (MVC) pattern. The Servlet lifecycle consists of the following steps:

- Servlet class is loaded by the container during start-up
- The container calls the init() method which initializes the Servlet and must be called before the Servlet can service any requests. The init() method is called only once in the entire lifecycle of a Servlet
- After initialization, servlet can process client requests
- The container calls the service() method of the Servlet for every request which determines the kind of request being made and sends it to an appropriate method to handle the request

B. UML Diagrams

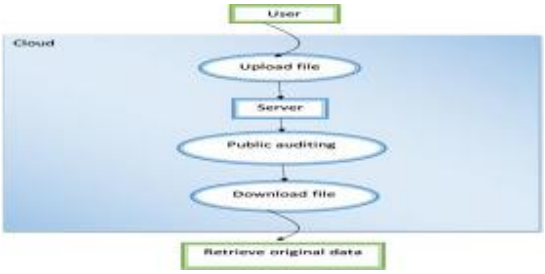

Fig. 4: DFD Level 0

- Finally, the container calls the destroy() method which takes the Servlet out of service. The destroy() method can also be called only once in the lifecycle of a Servlet similar to init() method.

JDBC - Java Database Connectivity (JDBC) is a framework for Java developers that access information stored in databases, spreadsheets, and flat files. JDBC is commonly used to connect a user program to

Unified Modeling Language (UML) is a standard language for visualizing, documenting, specifying and constructing the artifacts of software systems. UML can be represented as a general-purpose visual modeling language to specify, document, visualize and con- structs software system. Although UML is generally used to model software systems, it is not limited within this boundary.

C. Use Case Diagram

A Use Case is an explanation of a systems behavior from a user standpoint. For system developer, this is a valuable tool. it’s a standard technique for gathering system requirements from a user’s perspective. A little stick figure is used to recognize an actor and ellipse indicates use-case

D. Class Diagram

A class diagram shows the relationships and source code dependencies among classes in the UML. In this context, a class defines the variables and methods in an object, which is a specific entity in a program or unit of code that denotes that entity. Class diagrams are useful in all forms of object-oriented programming (OOP).

E. Sequence Diagram

It shows the interaction between a set of objects, through the messages that may be conveyed between them. The diagrams consist of interacting objects and actors, with messages in-between them it is common to focus the model on scenarios specified by use-cases.

F. Collaboration Diagram

A collaboration diagram, also called a communication diagram or interaction diagram, is an example of the relationships and interactions among software objects in the UML. The diagram mimics a flowchart that represent the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time.

G. Activity Diagram

Activity diagrams represent the workflow behavior of a system. Activities, branches (selections or conditions), transitions, forks and joins are the main elements in activity diagram. it is also used to research use case by narrates a complicated sequential algorithm, what actions need to take place, when they should occur, and modeling applications with parallel processes.

VI. SYSTEM IMPLEMENTATION

A. User Registration Module

For the registration of user with identity ID the group manager randomly selects a number. In the traceability phase, the group manager adds into the group user list. After the registration, user receives a private key which is utilized for group signature generation and file decryption.

B. Public Auditing Module

Homomorphic authenticators are unforgeable verification. To achieve privacy-preserving public auditing, we propose to uniquely integrate the Homomorphic authenticator with MD5 and SHA 256. Homomorphic authenticators are unforgeable verification metadata generated from individual data blocks, which can be se- curely accumulated in such a way to assure an auditor that a linear combination of data blocks is correctly computed by verifying only the aggregated authenticator. In our protocol, the linear combination of sampled blocks in the server's response is concealed with ran- domness created by a pseudo random function (PRF). The proposed scheme is as - Setup Phase and Audit Phase

C. Sharing Data Module

The canonical application is data sharing. The public auditing property is beneficial when we expect the delegation to be systematic and flexible. The schemes enable a content provider to share data in a confidential and selective way, with a fixed and small ciphertext expansion, by distributing to each authorized user a single and small aggregate key.

D. Integrity Checking Module

Supporting data dynamics for privacy-preserving public risk auditing is of at most importance. In this module, instead of auditor downloading all the user data and verifying information for a particular user, he/she can validate particular user information. This technique achieves privacy- preserving public risk auditing with support of data dynamics including block level operations of modification, deletion and insertion.

VII. CONCLUSION

- We proposed a novel privacy-preserving mechanism that sup- ports public auditing on shared data stored in the cloud based on blockchain to mainly achieve trustworthy data integrity as well as the recovery.
- For the future research, our work will be towards how to avoid this type of re-computation introduced by dynamic groups while still preserving identity privacy from the public verifier during the process of public auditing on shared data.

REFERENCES

[1] B. Wang, B. Li, and H. Li, "Certificateless Public Auditing for Data Integrity in the Cloud," Proc. IEEE Conf. Comm. and Network Security (CNS'13), pp. 276- 284, 2013

[2] B. Wang, B. Li and H. Li, "Public Auditing for Shared Data with Efficient User Revocation in the Cloud", Proc. IEEE INFOCOM, pp. 2904-2912, 2013

[3] C. Wang, S.S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy – Preserving Public Auditing for secure cloud storage", IEEE Trans. Computers, vol. 62, no. 2, pp. 362-375, Feb 2013

[4] Cong Wang, Student Member, IEEE, Sherman S.-M. Chow, Qian Wang, Student Member, IEEE, Kui Ren, Member, IEEE, and Wenjing Lou, Member, IEEE]

[5] Efficient and Secure Multi-Keyword Search on Encrypted Cloud Data (Y. Prasanna, Ramesh)

[6] Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud (Boyang Wang, Baochun Li and Hui Li State Key Laboratory of Inte- grated Services Networks, Xidian University, Xi'an, China Department of Electrical and Computer Engineering, University of Toronto, Toronto,Canada

[7] Panda: Public Auditing for Shared Data with Efficient User Revocation in the Cloud. (Boyang Wang, Baochun Li, Member, IEEE, and Hui Li, Member, IEEE)

[8] Remote Data Checking for Network Coding-based Distributed Storage Systems (Bo Chen, Reza Curtmola Department of Computer Science New Jersey Institute of Technology, Giuseppe Ateniese, Randal Burns Department of Computer Science Johns Hopkins University

[9] Short Group Signatures (Dan Boneh, Stanford University, Xavier Boyen, and Hovav Shacham, Stanford University)

[10] Storing Shared Data on the Cloud via Security-Mediator (Boyang Wang, Sherman S.M. Chow, Ming Li, and Hui Li State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China Department of Information Engineering, Chinese University of Hong Kong, Hong Kong Department of Computer Science, Utah State University, Logan, Utah, USA)

[11] The MD5 Message-Digest Algorithm (RFC1321), 2014