International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Ensuring Data Security in Cloud Using an Efficient Merkel Based Data Auditing Protocol

Monisha B[1], Dr. Girisan E K[2]

[1]III B.sc Computer Technology, Department Of Computer Technology
[2]Associate Professor and Head, Department Of Computer Technology
Department Of Computer Technology, Sri Krishna Adithya College of Arts and Sri Krishna Adithya College of Arts And Science, Coimbatore.

ABSTRACT

With a broad application of cloud storage, users can get many suitability such as low data remote storage and flexible data sharing. Considering the Cloud Services Provider (CSP), a lot of cloud auditing plans are proposed to ensure shared data security and integrity. To address the above problem, existing researchers prepared an efficient sampling verification algorithm. Based on this algorithm, the auditing scheme is further adapted, and a dynamic auditing function has been developed for the plan to facilitate data owners to update the data. However, this existing cloud auditing schemes have some security risks, such as user identity disclosure, service attacks refusal and single-manager of power abuse. In this work we propose the Merkel -based message certification code (MAC) that the audit cloud data storage in public is able to help fully establish this new born cloud economy. With public auditability, a reliable unit data owners with expertise and capabilities do not have the external audit party to assess the risk of outsourced data when necessary. Such an auditing service not only helps the data owners to save the calculation resources, but also provides a transparent yet a transparent for data owners to gain confidence in the cloud. Thus we increased the customer/user interaction between Swami and Cloud Server in this concept. We describe the approach and system requirements that must be brought into mind, and underline the challenges that need to be resolved to become a reality for such publicly audible safe cloud storage service.

1. INTRODUCTION

We begin with a high-level architecture description of cloud data storage services illustrated. At its core, the architecture consists of four different entities:

- Data owner
- User
- Cloud server (CS) and
- TPA.

Here the **TPA** is “the trusted entity that has expertise and capabilities to assess cloud storage security on behalf of a data owner upon request”. Under the cloud paradigm, the **data owner** “may represent either the individual or the enterprise customer, who relies on the cloud server for remote data storage and maintenance, and thus is relieved of the burden of building and maintaining local storage infrastructure”.

• Availability (capable of accessing data from anywhere),

• Relative low cost (paid as a function of requirement) and

• When sharing demand between a group of reliable users, such as a partner in the cooperation team or employee in the enterprise organization.

For simplicity, we consider single writer/many readers here. Only the data owner can dynamically interact with CS to update its stored data, while users only have the privilege of reading the file.

In the purview of this project, "We focus on how to publicly ensure audio safe data storage services". Since the data owner no longer has physical control of data, it is important to allow the data owner to verify that his data is correctly stored and maintained in the cloud.In the context of resources and expertise, possibly taking into account the large cost, the data owner may resort to data auditing work to ensure the storage safety of its data to the data owner TPA, hoping to keep the data private from the TPA .We believe that TPA, which is in the auditing business, is reliable and independent, and thus there is no incentive to collide with CS or owners during the auditing process.

TPA should be able to efficiently audit cloud data storage without local copy of data and for data owners without any additional online burden. In addition, any possible leakage of an owner's outsourced data towards TPA through the auditing protocol should be banned. We consider both malicious outdoor people and a semi-trans-on-as potential opponents that disrupt cloud data storage services ".Mainly outdoor outsiders can be economically motivated, and "the cloud storage is the ability to attack the server and subsequently pollute or remove the data of the owners while remaining undetermined".

The CS is in this sense semi-pearcruma, "Most of the time it behaves properly and does not distract from the prescribed protocol execution". However, CS can rarely ignore data files related to normal cloud owners for their own benefits, rarely ignoring or intentionally removing accessed data files. In addition, CS may decide to hide the data corruption caused by server hack or Byzantine failures to maintain its reputation.

Note that in our architecture, we believe that basic security mechanisms such as a preloaded public/private key pair "already have space to provide basic communication protection with each unit, which can be obtained in practice with a little overhead".

2. OBJECTIVE

Input design is the process of changing the user-oriented details of the input in a computer-based system. This design data is important to show the correct direction to the management to avoid errors in the input process and get the correct information from the computerized system. It is obtained by creating a user friendly screen for data entry to handle large amounts of data. The goal of designing input is to make data entry easier and be free from errors. The data entry screen is designed in such a way that all data can be manipulated. It also provides record viewing facility.It will investigate for its validity when the data is recorded. Data can be recorded with the help of screen. Appropriate messages are provided when needed so that the user is not immediately in maize. Thus the purpose of the input design is to create an input layout that is easy to follow.

3. SYSTEM STUDY

System analysis will be performed to determine if it is flexible to design information based on policies and plans of organization and on user requirements and to eliminate the weakness of present system. This chapter discusses the existing system, proposed system and highlights of the system requirements.

3.1 EXISTING SYSTEM

In this existing scheme, in which TPA acts by the working nodes in the smart contract to resist the collusion attack between CSP and TPA. Although this scheme is also based on the auditing framework instead of using the traditional random sampling method, here they design a novel ***sampling verification algorithm*** to improve the efficiency of the scheme. This scheme guarantees that if the data stored on the cloud is lost, DO will not need to pay a service fee, on the contrary, CSP needs to pay compensation to DO.

The main contributions of this efficient data auditing scheme are as follows:

- The size of the challenge set should be chosen by DO in advance, and it will be written into the smart contract after reaching an agreement between DO and CSP, which can resist the replacement attack from CSP.
- Designed a new sampling verification algorithm to ensure that there is no intersection between any adjacent challenge sets in two audits, which collecting corrupted blocks together on a small scale to increase the probability of successful detection.
- Developed the functions of data privacy-preserving, dynamic auditing, and batch auditing.

DISADVANTAGES OF EXISTING SYSTEM

- Because the amount of cloud data can be huge, it would be quite impractical for a data owner to retrieve all of her data just in order to verify the data is still correct.
- If the data auditing task is delegated to a TPA, this method inevitably violates our suggested requirements, with large auditing cost for a cloud server (for accessing and transferring all of the data) and data privacy exposure to the TPA (for retrieving a local copy of data).

4.2 PROPOSED SYSTEM

Utilizing Merkel based Message Authentication Codes (MACs) to significantly reduce the arbitrarily large communication overhead for public auditability without introducing any online burden on the data owner, we resort to this authenticator technique. Here authenticators are unforgeable metadata generated from individual data blocks, which can be securely aggregated in such a way to assure a verifier that a linear combination of data blocks is correctly computed by verifying only the aggregated authenticator. Using this technique requires additional information encoded along with the data before outsourcing.

ADVANTAGES:

It achieves a constant communication overhead for public auditability.

- Its excellent flexibility makes it applicable not only to large IT Companies and Internet companies, but also to medium-sized and even small IT systems.

The individual data operation on any file block, especially block insertion and deletion, will no longer affect other unchanged blocks.

5. SYSTEM SPECIFICATION

5.1 HARDWARE SPECIFICATION

Processor	:	i3 and above
RAM	:	2 GB and above
HDD	:	500GB and Above

5.2 SOFTWARE SPECIFICATION

Operating System	: Windows XP/8/10
Programming Language	: Java
Back End	: MS Access
IDE	: My Eclipse 6.0.

6. SYSTEM TESTING

Software testing is a critical element if software quality assurance represents the ultimate reviews of specification, design and coding. Testing is vital of the system.

Errors can be injected at any stage during development. During testing, the program is executed with correctness. A series of testing are performed for the proposed systems before the system is delivered to the user.

6.1 Unit Testing

In the unit testing the testing is performed on each module and this module is known as module testing. This testing was carried out during programming state itself. In this testing all the modules working satisfactorily as regard to the expected output from the module. Unit testing is a method by which individual units of source code are tested to determine if they are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual function or procedure. Unit tests are created by programmers or occasionally by white box testers.

Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviours that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development. Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit and how to use it can look at the unit tests to gain a basic understanding of the unit API.

6.2 Acceptance Testing

Acceptance testing is black-box testing performed on a system (e.g. software, lots of manufactured mechanical parts, or batches of chemical products) prior to its delivery. It is also known as functional testing, black-box testing, release acceptance, QA testing, application testing, confidence testing, final testing, validation testing, or factory acceptance testing.

Acceptance testing generally involves running a suite of tests on the completed system. Each individual test, known as a case, exercises a particular operating condition of the user's environment or feature of the system, and will result in a pass or fail, or Boolean, outcome. There is generally no degree of success or failure. The test environment is usually designed to be identical, or as close as possible, to the anticipated user's environment, including extremes of such. These test cases must each be accompanied by test case input data or a formal description of the operational activities (or both) to be performed—intended to thoroughly exercise the specific case—and a formal description of the expected results.

Types of Acceptance Testing

Typical types of acceptance testing include the following

User acceptance testing

This may include factory acceptance testing, i.e. the testing done by factory users before the factory is moved to its own site, after which site acceptance testing may be performed by the users at the site.

Operational acceptance testing

Also known as operational readiness testing, this refers to the checking done to a system to ensure that processes and procedures are in place to allow the system to be used and maintained.

Contract and regulation acceptance testing

In contract acceptance testing, a system is tested against acceptance criteria as documented in a contract, before the system is accepted. In regulation acceptance testing, a system is tested to ensure it meets governmental, legal and safety standards.

Alpha and beta testing

Alpha testing takes place at developers' sites, and involves testing of the operational system by internal staff, before it is released to external customers. Beta testing takes place at customers' sites, and involves testing by a group of customers who use the system at their own locations and provide feedback, before the system is released to other customers. The latter is often called “field testing”.

6.3 Integration Testing

One module can have adverse effect on another such functions when combined may not produce the desired results. Integration testing is a systematic technique for constructing the program structure and conducting test to uncover errors associated with interface. All the modules are combined in this testing step. The entire program is tested as the whole. The errors uncovered are corrected for the next testing step.

6.4 Black Box Testing

The black box approach is attesting method in which test data are delivered from the functional requirement without regard to the final program structure. Because only functionality of the software is concerned. In black box testing, only the functionality is determined by observing the outputs to the corresponding input. In this testing various input images are exercised and the output images are compared as required by the content retriever.

6.5 White Box Testing

White box testing are the software predicates on close examination of procedure details. It provides test cases that exercise specific test for conditions and loops. White box testing was carried out in the order to guarantee that

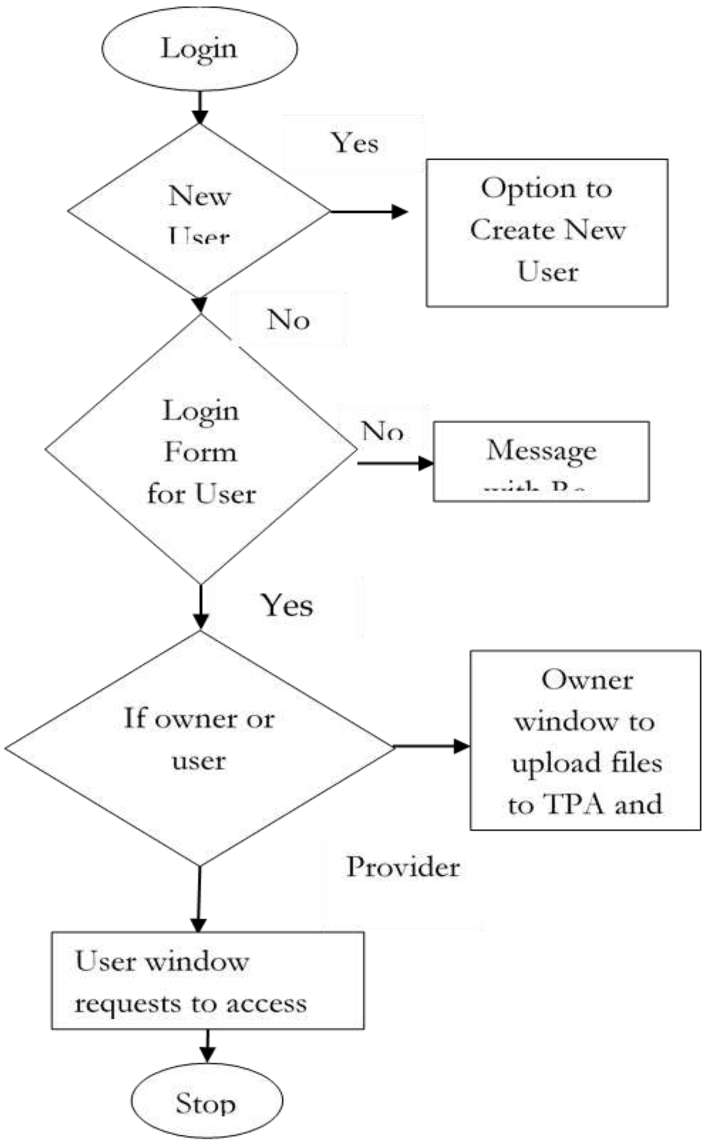
- All independent parts within a module exercised at least once.
- All logical decision on this true and false side was exercised

6.6 Validation Testing

Computer input procedures are designed to detect errors in the data at the lower level of detail which is beyond the capability of the control procedures. The validation succeeds when the software functions in the manner that can be reasonably expected by the customer.

7. SYSTEM ANALYSIS AND DESIGN

7.1 DATA FLOW DIAGRAM


Login
Yes
New User
Option to Create New User
No
Login Form for User
No
Message
Yes
If owner or user
Owner window to upload files to TPA and
Provider
User window requests to access
Stop

7.2 MODULE DESIGN

Minimize auditing overhead:

First and foremost, the overhead imposed on the cloud server by the auditing process must not outweigh its benefits. Such overhead may include both the I/O cost for data access and the bandwidth cost for data transfer. Any extra online burden on a data owner should also be as low as possible. Ideally, after auditing delegation, the data owner should just enjoy the cloud storage service while being worry-free about storage auditing correctness.

Protect Data Privacy:

Data privacy protection has always been an important aspect of a service level

Agreement for cloud storage services. Thus, the implementation of a public auditing protocol should not violate the owner's data privacy.

In other words a TPA should be able to efficiently audit the cloud data storage without demanding a local copy of data or even learning the data content.

Support Data Dynamics:

As a cloud storage service is not just a data warehouse, owners are subject to dynamically updating their data via various application purposes. The design of auditing protocol should incorporate this important feature of data dynamics in Cloud Computing.

Support Batch Auditing:

The prevalence of large-scale cloud storage service further demands auditing efficiency. When receiving multiple auditing tasks from different owners' delegations, a TPA should still be able to handle them in a fast yet cost-effective fashion.

This property could essentially enable the scalability of a public auditing service even under a storage cloud with a large number of data owners.

7.3 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:'

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.

Methods for preparing input validations and steps to follow when error occur.

7.4 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

7.5 DATABASE DESIGN

Databases are normally implemented by using a package called a Data Base Management System (DBMS). Each particular DBMS has somewhat unique characteristics, and so such, general techniques for the design of database are limited. One of the most useful methods of analyzing the data required by the system for the data dictionary has developed from research into relational database, particularly the work of E.F.Codd. This method of analyzing data is called "Normalization". Unnormalized data are converted into normalized data by three stages. Each stage has a procedure to follow.

NORMALIZATION:

The first stage is normalization is to reduce the data to its first normal form, by removing repeating items showing them as separate records but including in them the key fields of the original record. The next stage of reduction to the second normal form is to check that the record, which one is first normal form, all the items in each record are entirely dependent on the key of the record. If a data item is not dependent on the key of the record, but on the other data item, then it is removed with its key to form another record. This is done until each record contains data items, which are entirely dependent on the key of their record.

The final stage of the analysis, the reduction of third normal form involves examining each record, which one is in second normal form to see whether any items are mutually dependent. If there are any item there are removed to a separate record leaving one of the items behind in the original record and using that as the key in the newly created record.

BUSINESS MODELING:

The information flow among business function is modelled in a way that answers the following questions: what information drives the business process? What information is generated? What generate it? Where does the information go? Who process it?

DATA MODELING:

The information flow defined as a process of the business modelling is refined into a set of data objects that are needed to support the business. The characteristics (called attributes) of each object are identified and relationships between these objects are defined.

PROCESS MODELING:

The data objects defined in the data-modelling phase are transformed to achieve the information flow necessary to implement a business function. Processing description is created for addition, modifying, deleting, or retrieving a data object.

8. IMPLEMENTATION

The implementation phase focuses how the engineer attempts to develop the system. It also deals with how data are to be structured, how procedural details are to be implemented, how interfaces are characterized, how the design will be translated into programming and hoe the testing will be performed. The methods applied during the development phase will vary but three specific technical tasks should always occur.

- The software design
- Code generation
- Software testing

The system group has changed with responsibility to develop a new system to meet requirements and design and development of new information system. The source of these study facts is variety of users at all level throughout the organization.

8.1 Stage of Development of a System

- Feasibility assessment
- Requirement analysis
- External assessment
- Architectural design
- Detailed design
- Coding
- Debugging
- Maintenance

8.2 Feasibility Assessment

In Feasibility this stage problem was defined. Criteria for choosing solution were developed, proposed possible solution, estimated costs and benefits of the system and recommended the course of action to be taken.

8.3 Requirement Analysis

During requirement analysis high-level requirement like the capabilities of the system must provide in order to solve a problem. Function requirements, performance requirements for the hardware specified during the initial planning were elaborated and made more specific in order to characterize features and the proposed system will incorporate.

8.4 External Design

External design of any software development involves conceiving, planning out and specifying the externally observable characteristic of the software product. These characteristics include user displays, report formats, external data source and data links and the functional characteristics.

8.5 Internal Design Architectural and Detailed Design

Internal design involved conceiving, planning out and specifying the internal structure and processing details in order to record the design decisions and to be able to indicate why certain alternations were chosen in preference to others. These phases also include elaboration of the test plans and provide blue prints of implementation, testing and maintenance activities. The product of internal design is architectural structure specification.

The work products of internal design are architectural structure specification, the details of the algorithm, data structure and test plan. In architectural design the conceptual view is refined.

8.6 Detailed Design

Detailed design involved specifying the algorithmic details concerned with data representation, interconnections among data structures and packaging of the software product. This phase emphasizes more on semantic issues and less synthetic details.

8.7 Coding

This phase involves actual programming, i.e., transacting detailed design into source code using appropriate programming language.

8.8 Debugging

This stage was related with removing errors from programs and making them completely error free.

8.9 Maintenance

During this stage the systems are loaded and put into use. They also get modified accordingly to the requirements of the user. These modifications included making enhancements to system and removing problems.

9. CONCLUSION AND FUTURE ENHANCEMENT

Cloud computing has been envisioned as the next-generation architecture of enterprise IT. In contrast to traditional enterprise IT solutions, where the IT services are under proper physical, logical, and personnel controls, cloud computing moves the application software and databases to servers in large data centres on the Internet, where the management of the data and services are not fully trustworthy.

This unique attribute raises many new security challenges in areas such as software and data security, recovery, and privacy, as well as legal issues in areas such as regulatory compliance and auditing, all of which have not been well understood.

In this work we focus on cloud data storage security. We first present network architecture for effectively describing, developing, and evaluating secure data storage problems. We then suggest a set of systematically and MAC based cryptographically desirable properties for public auditing services of dependable cloud data storage security to become a reality. Through in-depth analysis, some existing data storage security building blocks are examined. Thus finally we proved our proposed algorithm overcome the limitation of existing.

SCOPE FOR FUTHURE ENHANCEMENT

In the above sections we have described some suggested requirements for public auditing services and the state of the art that fulfils them. However, this is still not enough for a publicly auditable secure cloud data storage system, and further challenging issues remain to be supported and resolved. We believe security in cloud computing, an area full of challenges and of paramount importance, is still in its infancy now but will attract enormous amounts of research effort for many years to come.

10. REFERENCE:

[1] M. Armrest et al., ``above the clouds: A Berkeley view of cloud computing," Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCBEECS-2009-28, 2009.

[2] Cloud Security Alliance. (2018). Top Threats to Cloud Computing: Deep Dive. [Online].

http://www.cloudsecurityalliance.org

[3] J. Xu and E.-C. Chang, ``towards efficient proofs of irretrievability," in Proc. Inf., Computer. Common. secure. 2012, pp. 7980.

[4] K. Ren, Wang, and Wang, ``Security challenges for the public cloud," IEEE Internet Computer., vol. 16, no. 1, pp. 6973, Jan. /Feb. 2012.

[5] Wang, X. Chen, X. Huang, I. You, and Y. Xiang, ``Verifiable auditing for outsourced database in cloud computing," IEEE Trans. Computer., vol. 64, no. 11, pp. 32933303, Nov. 2015.

[6] F. Chen, T. Xiang, Y. Yang, and S. S. M. Chow, ``Secure cloud storage meets with secure network coding," IEEE Trans. Computer., vol. 65, no. 6, pp. 19361948, Jun. 2016.

[7] H. Sachem and B. Waters, ``Compact proofs of irretrievability," in Proc. Int. Conf. Theory Appl. Crypto. Inf. Secure., 2008, pp. 90107.

[8] K. Yang and X. Jian, ``an efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Trans. Parallel Diatribe. Syst., vol. 24, no. 9, pp. 17171726, Sep. 2013.