Double Encryption Based Auditing Protocol Using Dynamic Operation in Cloud Storage

M.Mahindha
Computer science and Engineering,
UCE-Thirukkuvalai,
Nagapattinum,India
mahi201396@gmail.com

S.Muthu
Computer science and Engineering,
UCE-Thirukkuvalai,
Nagapattinum,India
muthu77796@gmail.com

R.Ranjitha
Computer science and Engineering,
UCE-Thirukkuvalai,
Nagapattinum,India
ranjitharam1996@gmail.com

Mr.D.Maria manuel vianny,
Department Of CSE,
UCE-Thirukkuvalai
Nagappattinum, India
viannycse@gmail.com

ABSTRACT: Using Cloud Storage, users can tenuously store their data and enjoy the on-demand great quality applications and facilities from a shared pool of configurable computing resources, without the problem of local data storage and maintenance. However, the fact that users no longer have physical possession of the outsourced data makes the data integrity protection in Cloud Computing a formidable task, especially for users with constrained dividing resources. From users' perspective, including both individuals and IT systems, storing data remotely into the cloud in a flexible on-demand manner brings tempting benefits: relief of the burden for storage management, universal data access with independent geographical locations, and avoidance of capital expenditure on hardware, software, and personnel maintenances, etc. . To securely introduce an effective third party auditor (TPA), the following two fundamental requirements have to be met: 1) TPA should be able to capably audit the cloud data storage without demanding the local copy of data, and introduce no additional on-line burden to the cloud user; 2) The third party auditing process should take in no new vulnerabilities towards user data privacy. In this project, utilize and uniquely combine the public auditing protocols with double encryption approach to achieve the privacy-preserving public cloud data auditing system, which meets all integrity checking without any leakage of data. To support efficient handling of multiple auditing tasks, we further explore the technique of online signature to extend our main result into a multi-user setting, where TPA can perform multiple auditing tasks simultaneously. We can implement double encryption algorithm encrypt the data twice and stored cloud server.

INDEX TERMS *– Cloud Storage, Privacy-Preserving, Double Encryption, Public Auditing, Online/Offline signature, Batch Auditing*

*****

I. INTRODUCTION

CLOUD COMPUTING

Cloud computing is a computing paradigm, where a large pool of systems are connected in private or public networks, to provide dynamically scalable with the advent of this technology, the cost of computation, application hosting, content storage and delivery is reduced significantly. It is a practical approach to experience direct cost benefits and it has the potential to transform a data center from a capital-intensive set up to a variable priced environment. The idea of cloud computing is based on a very fundamental principles of reusability of IT capabilities. The difference that cloud computing brings compared to traditional concepts of "grid computing", "distributed computing", "utility computing", or "autonomic computing" is to broaden horizons across organizational boundaries. Forrester [1] defines cloud computing as: "A pool of abstracted, highly scalable, and managed compute infrastructure capable of hosting end customer applications and billed by consumption". It is a technology that uses the internet and central remote servers to maintain data and applications and allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access. This technology allows for much more efficient computing by centralizing data storage, processing and bandwidth. Cloud computing examples are Yahoo email, Gmail, or Hotmail.

II. ARCHITECTURE

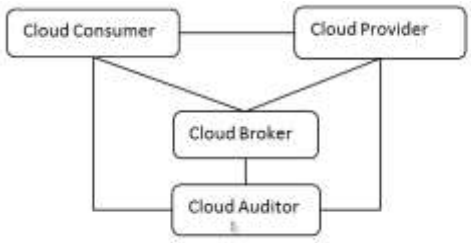


Figure 1.1 Architecture of cloud computing

Cloud Provider
A person, organization, or entity responsible for making a service available to interested parties. A Cloud Provider acquires and manages the computing infrastructure required for providing the services, runs the cloud software that provides the services, and makes arrangement to deliver the cloud services to the Cloud Consumers through network access.

Primary Cloud Provider
A Primary Provider offers services hosted on infrastructure that it owns. It may make these services available to Consumers through a third party (such as a Broker or Intermediary Provider), but the defining characteristic of a Primary Provider is that it does not source its service offerings from other Providers.

Cloud Consumer
"A person or organization that maintains a business relationship with, and uses service from, Cloud Providers. A cloud consumer browses the service catalog from a cloud provider, requests the appropriate service, sets up service contracts with the cloud provider, and uses the service. The cloud consumer may be billed for the service provisioned, and needs to arrange payments accordingly." What is not covered here is the **end user** that consumes the possibly enriched service offered by the Cloud Consumer. In SaaS, the Cloud Consumer is often identical with the end user. However, in business environments this is not always the case. Using the example of GMail, only the paying entity is the Cloud Customer (e.g. IT department) while many other employees may use the mailing service as end users.

Cloud Auditor
"As cloud computing evolves, the integration of cloud services can be too complex for cloud consumers to manage. A cloud consumer may request cloud services from a cloud broker, instead of contacting a cloud provider directly. Hence the broker is an entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers." Brokers provide three different types of services to the Cloud Consumer.

Mediating Broker
A cloud broker enhances a given service by improving some specific capability and providing value-added services to cloud consumers. The improvement can be managing access to cloud services, identity management, performance reporting, enhanced security, etc.

Aggregating Broker
A cloud broker combines and integrates multiple services into one or more new services. The broker provides data integration and ensures the secure data movement between the cloud and multiple cloud providers.

Arbitraging Broker
Service arbitrage is similar to service aggregation except that the services being aggregated are not fixed. Service arbitrage means a broker has the flexibility to choose services from multiple agencies. The cloud broker, for example, can use a credit-scoring service to measure and select an agency with the best score.

III. PRIVACY PRESERVING

While the storage of corporate data on remote servers is not a new development, current expansion of cloud computing justifies a more careful look at its actual consequences involving privacy and confidentiality issues. 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. In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network. Besides, it is often insufficient to detect the data corruption only when accessing the data, as it does not give users correctness assurance for those un-accessed data and might be too late to recover the data loss or damage. To fully ensure the data integrity and save the cloud users' computation resources as well as online burden, it is of critical importance to enable public auditing service for cloud data storage, so that users may resort to an independent third party auditor (TPA) to audit the outsourced data when needed. The TPA, who has expertise and capabilities that users do not, can periodically check the integrity of all the data stored in the cloud on behalf of the users, which provides a much more easier and affordable way for the users to ensure their storage correctness in the cloud. In a word, enabling public auditing services will play an important role for this nascent cloud economy to become fully established; where users will need ways to assess risk and gain trust in the clou

IV. RELATED WORK

While cloud computing makes various advantages, it can be mentioned in chapter 1 and challenging security threats toward users' outsourced data. Since cloud service providers (CSP) are separate administrative entities, data outsourcing is actually relinquishing user's ultimate control over the fate of their data. As a result, the correctness of the data in the cloud is being put at risk due to the following reasons First of all, although the infrastructures under the cloud are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity. Examples of outages and security breaches of noteworthy cloud services appear from time to time. Second, there do exist various

motivations for CSP to behave unfaithfully toward the cloud users regarding their outsourced data status. CSP might reclaim storage for monetary reasons by discarding data that have not been or are rarely accessed, or even hide data loss incidents to maintain a reputation. In short, although outsourcing data to the cloud is economically attractive for long-term large-scale storage, it does not immediately offer any guarantee on data integrity and availability. This problem, if not properly addressed, may impede the success of cloud architecture. 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. In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network. Besides, it is often insufficient to detect the data corruption only when accessing the data, as it does not give users [2] correctness assurance for those un-accessed data and might be too late to recover the data loss or damage.

Disadvantages
Leak users' data to external auditor. Can extract the original data of a user during the auditing process. Existing system provide insecurity scheme for data auditing. Provide computational overheads

V. IMPLEMENTATION

The system model in this project involves three parties: the cloud server, a group of users and a public verifier. There are two types of users in a group: the original user and a number of group users. The original user initially creates shared data in the cloud, and shares it with group users. Both the original user and group users are members of the group. Every member of the group is allowed to access and modify shared data. Shared data and its verification metadata (i.e. signatures) are both stored in the cloud server. A public verifier, such as a third-party auditor (TPA) providing expert data auditing services or a data user outside the group intending to utilize shared data, is able to publicly verify the integrity of shared data stored in the cloud server. When a public verifier wishes to check the integrity of shared data, it first sends an auditing challenge to the cloud server. After receiving the auditing challenge, the cloud server responds to the public verifier with an auditing proof of the possession of shared data. Then, this public verifier checks the correctness of the entire data by verifying the correctness of the auditing proof. Essentially, the process of public auditing is a challenge and-response protocol between a public verifier and the cloud server.

Public AuditingA public verifier is able to publicly verify the integrity of shared data without retrieving the entire data from the cloud.
Correctness A public verifier is able to correctly verify shared data integrity.

Unforgetability Only a user in the group can generate valid verification metadata (i.e., signatures) on shared data.

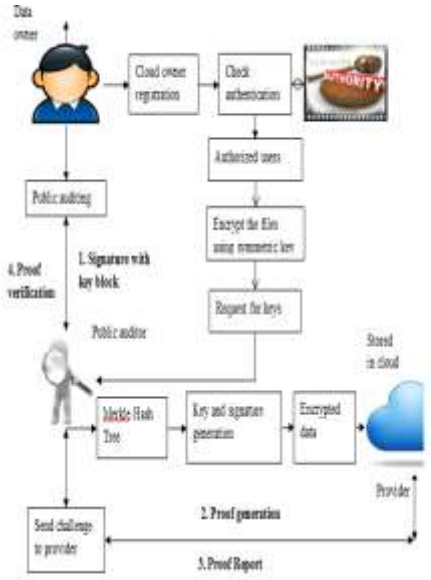
Identity PrivacyA public verifier cannot distinguish the identity of the signer on each block in shared data during the process of auditing. With cloud computing and storage, users are able to access and to share resources offered by cloud service providers at a lower marginal cost. It is routine for users to leverage cloud storage services to share data with others in a group, as data sharing becomes standard feature in most cloud storage offerings, including Dropbox, iCloud and Google Drive. The integrity of data in cloud storage, however, is subject to skepticism and scrutiny, as data stored in the cloud can easily be lost or corrupted due to the inevitable hardware/software failures and human errors. 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 or hash values of the entire data. Certainly, this conventional approach able to successfully check the correctness of cloud data. However, the efficiency of using this traditional approach on cloud data is in doubt. The main reason is that the size of cloud data is large in general. Downloading the entire cloud data to verify data integrity will cost or even waste user's amounts of computation and communication resources, especially when data have been corrupted in the cloud. Recently, many mechanisms have been proposed to allow not only a data owner itself but also a public verifier to efficiently perform integrity checking without downloading the entire data from the cloud, which is referred to as public auditing. In these mechanisms, data is divided into many small blocks, where each block is independently signed by the owner; and a random combination of all the blocks instead of the whole data is retrieved during integrity checking. A public verifier could be a data user (e.g. researcher) who would like to utilize the owner's data via the cloud or a third-party auditor (TPA) who can provide expert integrity checking service. In this proposed system we can implement Merkle Hash Tree to spilt the files into various parts and to provide double encryption concept to encrypt the data first at owner side and again encrypt the data based on TPA provided keys. Finally provide batch auditing schemes to perform multiple tasks at a time and user level privacy can be implemented to share the data without any leakages.

Advantages
Improved Public auditability and privacy-preserving. Fully data dynamics. Fast auditing and low performance protocols. End device friendliness.

VI. SYSTEM DESIGN

SYSTEM ARCHITECTURE DESIGN
In proposed system, we can implement public auditing scheme to monitor the data from modify attacks. Cloud owner can be authenticated by trusted third party. Authorized owner can be uploading the files in encrypted format. First encryption can be done using symmetric encryption algorithm. And then spilt the files into chunks.
Chunks are encrypted using Merkle hash tree algorithm. Encrypted files are uploaded to cloud storage and maintained by cloud server. The cloud owner can be send the auditing messages to cloud server through TPA. The messages can be send is in the form of online signature. Cloud server can be audit the data and to provide proofs to owner about the status of storage. The TPA can be

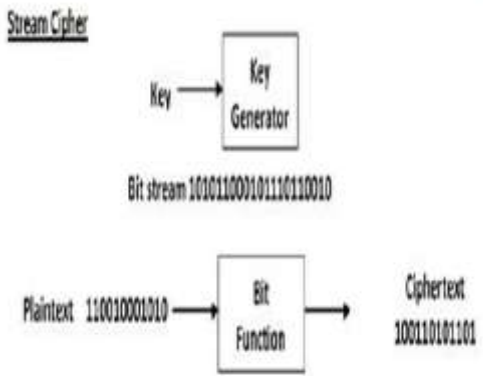
performing batch auditing scheme. Finally cloud users can be access the data from cloud with the permission of cloud owners.



VII. IMPLEMENTATION STAGES

Symmetric key algorithm

Astream cipheris a symmetric key cipher where plaintext digits are combined with a pseudorandom cipher digit stream (key-stream). In a stream cipher, each plaintext digit is encrypted one at a time with the corresponding digit of the key-stream, to give a digit of the cipher-text stream. Since encryption of each digit is dependent on the current state of the cipher, it is also known as state cipher. In practice, a digit is typically a bit and the combining operation an exclusive-or (XOR). The steps are



Merkle Hash Tree (MHT)

To achieve privacy-preserving public auditing, propose to uniquely integrate the linear authenticator with binary tree technique. In our protocol, the linear combination of sampled blocks in the server's response is masked with randomness generated by the server. With random masking, the TPA no longer has all the necessary information to build up a correct group of linear equations and therefore cannot derive the user's data content, no matter how many linear combinations of the same set of file blocks can be collected. On the other hand, the correctness validation of the block-authenticator pairs can still be carried out in a new way which will be shown shortly, even with the presence of the randomness. Our design makes use of a public key-based MHT, to equip the auditing protocol with public audit ability. A MHT Encryption scheme is comprised of a tuple of algorithms (Gen, E,D, Eval), and is defined with respect to a circuit C with t inputs. Though a MHT scheme can be either a public-key or symmetric-key system, we will define it as a public-key system here. The key generation algorithm Gen takes the security parameter 1^k as input, and outputs the public key and private key for the system (Notation: (pk, sk) ← Gen(1^k)).

Assume that messages M ∈ $\{0, 1\}^{l(k)}$.

The encryption algorithm E takes a public key and a message as input, and outputs a ciphertext C, (Notation: C ← E(pk,M) for M ∈ $\{0, 1\}^{l(k)}$).

The decryption algorithm D takes a secret key and a ciphertext, and returns a message, (Notation: M ← D(sk,C) and M ∈ $\{0, 1\}^{l}$).

Finally, the evaluation algorithm Eval takes as input a public key, a description of a t-input circuit C, and t ciphertexts C_1, . . . ,C_t such that Ci← E(pk,Mi), and produces as output C*, (Notation: C* ← Eval(pk,C, C_1, . . . ,C_t)).

We add a new correctness property to the standard correctness requirement for an encryption scheme as follows. We say that an encryption scheme is homomorphic with respect to a t-input circuit C if ∀k, ∀M1, . . . ,Mt, Pr[(pk, sk) ← Gen(1k); C1, . . . Ct ← E(pk,M1), . . . , E(pk,Mt); C* ← Eval(pk,C, C_1, . . . ,C_t) : D(sk,C*) = C(M_1, . . . ,M_t)] = 1.

Similarly, a scheme with respect to a family of circuits $\{C_i\}$ if the correctness property holds for any circuit C ∈ $\{C_i\}$. Note that so far, our definition makes no requirement that the output C* of Eval should look like a standard ciphertext. Indeed, without some additional restriction on C*, every standard encryption scheme (Gen, E,D) can be trivially modified to yield a homomorphic encryption scheme (Gen', E',D', Eval') with respect to all circuits as follows.

Gen' runs as Gen.

E' runs as E.

The Eval' is constructed to take a public key, a circuit description, and up to t ciphertexts, and then output the circuit description concatenated with each of the ciphertexts, as C* ← Eval"(pk,C, C1, . . . ,Ct) = C|C1| . . . |Ct, with | used to denote concatenation.

On special cipher texts C* containing a circuit description, D' parses its input into C, C_1, . . . ,C_t, runs the original decryption algorithm D on the ciphertexts to obtain messages Mi ← D(sk,Ci), and runs the circuit C on these messages, to obtain D'(sk,C*) = C(M_1, . . . ,M_t), satisfying the homomorphic correctness property. On ciphertexts withoutcircuit descriptions, D'(sk,C) simply returns D(sk,C).

Batch auditing

With the establishment of privacy-preserving public auditing, the TPA may concurrently handle multiple auditing upon different users' delegation. The individual auditing of these tasks for the TPA can be tedious and very inefficient. Given K auditing delegations on K distinct data files from K different users, it is more advantageous for the TPA to batch these multiple tasks together and audit at one time. Keeping this natural demand in mind, we slightly modify the protocol in a single user case, and achieve the aggregation of K verification equations (for K auditing tasks) into a single one. As a result, a secure batch auditing protocol for simultaneous auditing of multiple tasks is obtained.

Verify file tag tk for each user k, and quit if fail For each user k ($1 \leq k \leq K$)

Generate a random challenge

Compute $\mu k, \sigma k,$ Rkas single user case;

Chal = {(I, Vi)} i∈ I

Compute R=R1, R2,….Rk

L = vk1||vk2|| …..$||vk_k$

Compute $\mu k = rk + \gamma k \mu' k \mathrm{mod} p$

Compute $\gamma k = h(R||Vk||L)$ for each user k and do batch auditing

VIII. PERFORMANCE EVALUATION

System testing: Testing is the stage of implementation of which aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct the goal will be achieved. The candidates system subject to a variety of tests. Online response, volume, stress, recovery, security and usability tests. A series of testing are performed for the proposed system before the system is ready for user acceptance testing.

Unit Testing: The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated .Then the web form level is made.

Integration Testing: Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all its correct and an opportunity to show the user that the system works.

Validation Testing: The final step involves validation testing which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called "Alpha and Beta test" to uncover that only the end user seems able to find. The compilation of the entire project is based on the full satisfaction of the end users.

Acceptance Testing: Acceptance testing can be defined in many ways, but a simple definition is the succeeds when the software functions in a manner that can be reasonable expected by the customer. After the acceptance test has been conducted, one of the two possible conditions exists. This is to fine whether the inputs are accepted by the database or other validations. For example accept only numbers in the numeric field, date format data in the date field. Also the null check for the not null fields. If any error occurs then show the error messages. The function of performance characteristics to specification and is accepted. A deviation from specification is uncovered and a deficiency list is created.

White Box Testing: White box testing, sometimes called "Glass-box testing". Using white box testing methods, the following tests were made on the system.All independent paths with in a module have been exercised at least once.All logical decisions were checked for the true and false side of the values. All loops were executed to check their boundary values.

Black Box Testing: Black box testing focuses on the functional requirements of the software. That is black box testing enables the software engineer to drive a set of input conditions that will fully exercise the requirements for a program. Black box testing is not an alternative for white box testing techniques. Rather, it is a complementary approach that is likely to uncover different class of errors. Black box testing attempts to find errors in the following categories:

1 .Interface errors.

2. Performances in data structures or external database access.

3. Performance errors.

4. Initialization and termination errors.

5. Incorrect or missing functions.

IX. CONCLUSION

Cloud computing securities are discussed and analyzed in previous study. In this project, some of the privacy threats are addressed and the techniques to overcome them are surveyed. While some approaches utilized traditional cryptographic methods to achieve privacy, some other approaches kept them away and focused on alternate methodologies in achieving privacy. Also, approaches to preserve privacy at the time of public auditing are also discussed. Thus, to conclude it is necessary that every cloud user must be guaranteed that his data is stored, processed, accessed and audited in a secured manner at any time. Data freshness is essential to protect against misconfiguration errors or rollbacks caused intentionally and can develop an authenticated file system that supports the migration of an enterprise-class distributed file system into the cloud efficiently, transparently and in a scalable manner. It's authenticated in the sense that enables an enterprise tenant to verify the freshness of retrieved data while performing the file system operations. The user must be given complete access control over the published data. Also, powerful security mechanisms must always supplement every cloud application. Attaining all these would end up in achieving the long dreamt vision of secured Cloud Computing in the nearest future.

X. FUTURE WORK

In future, this proposed model could be used to get the secure cloud computing environment which would be a great enhancement in the privacy preservation. And implement various protocols to improve the security of the system.

REFERENCES

[1] G. Ateniese, A. Faonio, and S. Kamara, "Leakage-resilient identification schemes from zero-knowledge proofs of storage," in IMA Inte. Conf. Cryptography and Coding, 2015, pp. 311–328.

[2] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," IEEE Trans. Parallel Distrib. Syst., doi. 10.1109/TPDS. 2015.2506573.

[3] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," IEEE Trans. Knowl. Data Eng., vol.23, no.9, pp.1432-1437, 2011.

[4] H. Liu, L. Chen, Z. Davar, and M. Pour, "Insecurity of an efficient privacy-preserving public auditing scheme for cloud data storage," J. Universal Comput. Sci., vol. 21, no. 3, pp. 473–482, 2015.

[5] J. K. Liu, M. H. Au, X. Huang, R. Lu, and J. Li, "Fine-grained twofactor access control for web-Based cloud computing services," IEEE Trans. Inf. Forens. Security, vol. 11, no. 3, pp. 484–497, 2016