SECURED CLOUD STORAGE SYSTEM BASED ON PRIVACY PRESERVING WEIGHTED SIMILARITY KEYWORD SEARCH SCHEME

A.Bharathi[1], P.Chitra Devi[2], U.Sundhar[3], D.Kanthasamy[4]

[1]P.G student, Department of CSE, Thiruvalluvar College of Engineering and Technology, Vandavasi.

[2]Assistant Professor, Department of CSE, Thiruvalluvar College of Engineering and Technology Vandavasi

[3]Head of the Department, Department of CSE, Thiruvalluvar College of Engineering and Technology, Vandavasi.

[4]Head of the Department, Department of CSE(AI&ML),Thiruvalluvar College of Engineering and Technology Vandavasi.

ABSTRACT - *Now a day, cloud storage systems are widely utilized by people everywhere. They are primarily used for uploading files to cloud servers and retrieving them from anywhere around the globe. In this cloud storage environment, all users store their own files, which can be accessed by unauthorized users without the owner's permission. This poses a risk of misuse of the cloud files. Therefore, we implement encryption for the uploaded files before storing them on the cloud server. However, this method alone does not guarantee the security of files in the cloud storage system. In this paper, we will explore how to securely store files on cloud servers using multiple keys, which ensures that unauthorized users cannot access the files without these two keys. Additionally, we encrypt the keys and store them on the cloud server. When a user wishes to download a file from the cloud server, they must request the key from the file owner. The owner grants permission for access, allowing the user to retrieve the file. We utilize a key store specific to the cloud storage, with files stored across different nodes. When the owner grants permission to a user, they send the key in an image format. This image background is only visible to the file user. The file key is only revealed to the user when a request is made to the file owner. Subsequently, the user can download the file from the cloud server after decryption.*

Key words: cloud server, **Searchable Encryption, key-based access control, attribute-based access control**

1. INTRODUCTION

The cloud has emerged as a crucial platform for data storage and processing. Examples of cloud services encompass online file storage, social networking platforms, email services, and online business applications. It consolidates virtually unlimited resources and provides elastic services to end users without requiring them to manage their own systems or make upfront equipment purchases.

Nevertheless, data confidentiality protection (to conceal plaintext from the cloud server and other unauthorized users) and data access control (to assign user access privileges) are typically necessary for data owners to securely store their information in the cloud. Encryption is a widely adopted technique to maintain data confidentiality by saving ciphertext in the cloud.

However, this may render traditional methods designed for plaintext keyword searches ineffective. In pursuit of facilitating secure and efficient searches over encrypted data, Searchable Encryption (SE) has garnered increasing attention in recent years, where a query is encrypted as a search capability, allowing a cloud server to return files that match the capability without needing to know the keywords in either the capability or the file's encrypted index.

Nevertheless, most existing SE schemes operate under the assumption that users can access all shared files. This assumption is not valid in cloud environments, where users are assigned varying access permissions based on the access-control policies set by data owners. Consequently, it is crucial to explore methods for effectively enforcing access control policies while searching through encrypted data. Numerous studies have addressed access control in the context of encrypted data. These studies can be divided into two categories:

key-based access control (KBAC) and attribute-based access control (ABAC). In KBAC, each file's decryption key is typically assigned directly to authorize users. As a user accumulates an increasing number of these keys, the burden of managing them can become excessive. To alleviate this burden, ABAC associates a set of attribute values with a user (or a file) and formulates an access policy for a file (or a user, respectively). A file is accessible only if the attribute values align with the access policy. Access keys, such as decryption keys in KBAC and keys representing attribute values in ABAC, must generally be kept confidential to safeguard data security from potential breaches.

Therefore, the traditional method for executing encrypted searches with access control involves carrying out search operations on the cloud server, leveraging its substantial computational capabilities, while the enforcement of access control is managed on users' devices to prevent the exposure of their access keys. This division between search execution and access control enforcement may result in performance issues, particularly when users possess varying access rights to search through different encrypted data stored in the cloud.

Need for the study

The investigation into a "Secured Cloud Storage System Based on Privacy-Preserving Weighted Similarity Keyword Search Scheme" is particularly pertinent in the current cloud computing environment, where the protection of data privacy, security, and effective data retrieval are of utmost importance.

Growing Use of Cloud Storage

As cloud computing services continue to expand, both individuals and organizations are increasingly depending on cloud storage for data management, backup, and sharing purposes. This trend has sparked concerns regarding the safeguarding of sensitive information while still enabling users to conveniently access and search for data. Cloud storage providers typically have access to user data, which heightens privacy issues. Many users express hesitation about storing sensitive data in the cloud due to their lack of control over the underlying physical infrastructure.

Keyword-Based Search

Keyword search enables users to swiftly locate pertinent information within extensive databases; however, conventional search methods necessitate plaintext access to the data, which jeopardizes privacy. In various situations, not all keywords hold the same significance. Weighted similarity search entails assigning varying levels of importance or relevance to each keyword, thereby improving search quality. A weighted search framework permits users to prioritize results according to the significance of the keywords. In this project, we introduced a novel AES encryption technique for storing files on a cloud server, aimed at enhancing the security of the cloud server.

Objective of the paper

The aim of this work is to create and evaluate a system that guarantees secure data storage and effective keyword-based search while safeguarding the privacy of both the data and the search queries. We aim to develop a cloud storage system that ensures the confidentiality and security of user data. Additionally, we will create a privacy-preserving keyword search mechanism that enables users to search for specific data without revealing the content of their queries to the cloud service provider.

2.LITERATURE REVIEW

In the paper titled "Efficient Secure Privacy Preserving Multi Keywords Rank Search over Encrypted Data in Cloud Computing" [1], a search method is proposed where Verifiable multi-keyword rank searchable encryption serves as a crucial technique for retrieving encrypted data from a cloud server while also validating the accuracy of the search results. Given that the cloud server may be malicious, it could potentially return incorrect search results or experience system faults. Consequently, designing a verifiable secure multi-keyword ranked search for outsourced encrypted data remains an unresolved issue. In this paper, we utilize T-SNE techniques to reduce dimensionality and transform features. We then implement a strategy for clustering similar keywords to create an enhanced inverted index. This is linked to coordinate matching techniques to enhance the accuracy of multi-keyword rank searches, thereby improving the user experience. Furthermore, we employ Paillier Homomorphic Encryption (P.H.E) and HMAC techniques for the encryption of the search index and verification of keyword searches, aiming to facilitate verifiable multi-keyword rank searches over encrypted data in the cloud server. The time required for ranking score computation and search efficiency can be enhanced through two methods: the idea of keyword grouping and the omission of dummy keywords in both queries and document vectors. As a result, our scheme achieves verifiable multi-keyword search with exact ciphertext retrieval. We assessed the performance of our proposed scheme using real-world datasets, ensuring sufficient security and privacy.

The paper titled "Enabling privacy-preserving multi-server collaborative search in smart healthcare" [2] discusses how the evolution of smart healthcare has led to medical institutions storing vast amounts of users' medical data on their cloud servers for purposes of diagnosis and treatment. However, the conventional storage framework presents challenges to the effective access to medical resources and the circulation of medical data, including issues related to data security, query privacy, and the creation of isolated data silos. A straightforward solution would be to consolidate all encrypted medical data within a central server. Nonetheless, this approach is heavily reliant on the central server, which introduces additional performance drawbacks and privacy concerns. Consequently, there is an urgent need to develop a secure and effective medical data retrieval system. In this paper, we propose a multi-server search framework that builds upon the existing data storage model to collaboratively facilitate the location of diagnostic institutions, medical data searches, and even cross-domain data searches. This multi-server architecture addresses the issues of destructiveness and excessive information centralization associated with a single server, thereby improving the system's reliability and practicality. By employing hidden vector encryption, secret sharing, and secure multi-party computation, we achieve efficient search capabilities, identity privacy, search pattern security, and access pattern security. Security analyses confirm that both identity privacy and query security are safeguarded. Comprehensive experiments indicate that our scheme demonstrates superior efficiency in data searching and data addition when compared through both horizontal and vertical analyses.

In this paper titled "Efficient Secure Privacy Preserving Multi Keywords Rank Search over Encrypted Data in Cloud Computing" [3], a search method is proposed that addresses issues such as processing overhead, data and keyword privacy, and minimizes both communication and computation costs. The data owner constructs an index that includes

relevance scores based on keyword frequency for the files. The user sends a request 'w' to the cloud server, optionally including 'k' as Tw, using a private key. The cloud server then searches the index using these scores and returns the encrypted file according to the ranked order. The searchable index is created using a Multidimensional B-tree. The owner generates an encrypted query vector $\bar{Q}$ for the set of file keywords. The user receives the corresponding encrypted query vector of W from the owner, which is then provided to the cloud server. The cloud server searches the index utilizing the Merkle–Damgård construction algorithm, compares the cosine measures of the file and query vectors, and returns the top k encrypted files to the user.

In the paper "Toward Secure Multikeyword Top k Retrieval over Encrypted Cloud Data" [5], a search method is introduced that employs Two Round Searchable Encryption (TRSE). In the first round, users submit multiple keywords 'REQ' 'W' as an encrypted query to achieve data and keyword privacy, creating a trapdoor (REQ, PK) as Tw, which is sent to the cloud server. The cloud server then calculates the scores from the encrypted index for the files and returns the encrypted score result vector to the user. In the second round, the user decrypts N using a secret key, calculates the file rankings, and requests the files with the top k scores. The file ranking occurs on the client side, while scoring is performed on the server side.

In the paper "Privacy-Preserving MultiKeyword Ranked Search over Encrypted Cloud Data" [4], a search method is proposed for a known ciphertext model and background model over encrypted data, which provides low computation and communication overhead. Coordinate matching is selected for multi-keyword searches. They utilize inner product similarity to quantitatively assess the similarity for file ranking. However, a drawback of MRSE is that it has a small standard deviation σ, which undermines keyword privacy.

In the paper "Protecting Your Right: Attribute based Keyword Search with Fine grained Owner enforced Search Authorization in the Cloud" [5], an Attribute-based Keyword Search is proposed that facilitates conjunctive keyword searches, ensures keyword semantic security, and provides Trapdoor unlinkability. The owners create an index containing all keywords along with an access list that includes policy attributes specifying the users authorized to perform searches. Subsequently, the owners encrypt the document and the index with the access list using a ciphertext policy attribute-based encryption technique. For user membership management, they employ proxy reencryption and lazy re-encryption techniques to distribute the workload to the cloud server (CS). The user sends a request for the trapdoor (Tw) to the CS using their private key. The CS then retrieves the Tw, searches the encrypted indexes, and returns files only if the user's attributes in the Tw meet the access policies in the indexes, resulting in coarse-grained dataset search authorization.

In the paper "Efficient similarity search over encrypted data" [6], a secure tree-based search scheme is suggested for encrypted cloud storage, which supports multi-keyword ranked searches along with dynamic operations on the document collection available at the server. The vector space model and the term frequency (TF) × inverse document frequency (IDF) model are combined in the construction of the index and the generation of queries to provide multi-keyword ranked search results. To achieve high search efficiency, the author constructs a tree-based index structure and proposes a Greedy Depth-first Search algorithm based on this index tree. Due to the unique structure of the tree-based index, the proposed search scheme can achieve sub-linear search times flexibly and effectively manage the

deletion and insertion of documents. The kNN algorithm is utilized to encrypt the index and query vectors, ensuring accurate relevance score calculations between the encrypted index and query vectors.

3.IMPLEMENTATION

A software application is generally developed after following the complete life cycle method of a paper. Various life cycle processes, including requirement analysis, design phase, verification, testing, and ultimately the implementation phase, contribute to effective project management. System implementation is a crucial phase where theoretical design is transformed into a practical system.

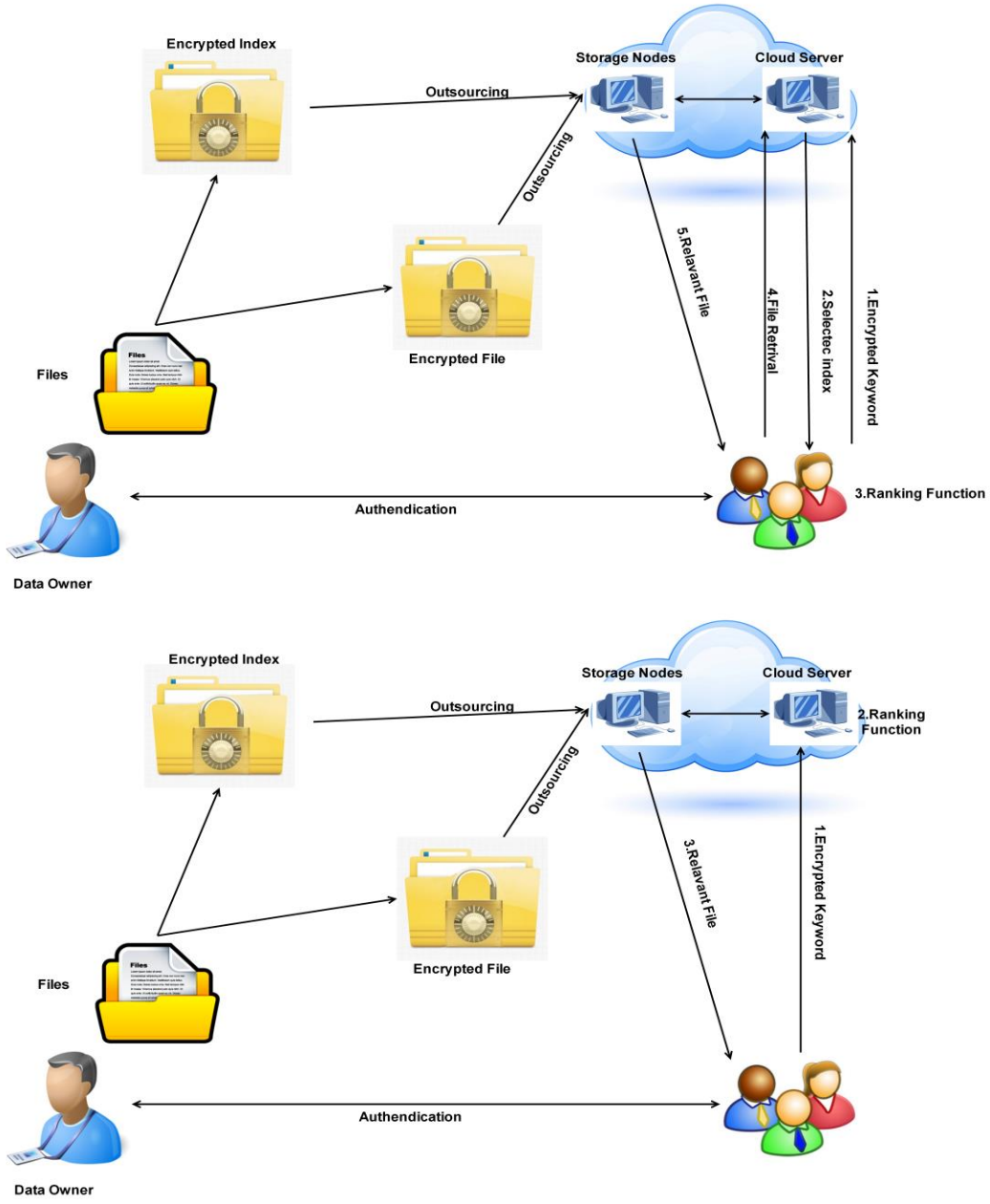

Fig.3.1 System Architecture

Implementation represents the point in the paper where the theoretical design is realized as a functioning system. Therefore, it is regarded as the most vital phase in ensuring the success of a new system and instilling confidence in the user that the new system will operate effectively. The implementation phase requires meticulous planning, an examination of the existing system and its constraints on implementation, the design of methods for changeover, and the assessment of changeover strategies.

Each program is tested individually during development using specific data, ensuring that the programs are interconnected as outlined in the program specifications. The computer system and its environment are evaluated to meet the user's satisfaction. Once the developed system is accepted and deemed satisfactory by the user, it is set to be implemented shortly. A straightforward operating procedure is provided to help the user quickly and clearly understand the various functions. The final step involves documenting the entire system, which includes the components and operating procedures of the system.

Provider

In this module, similar to a file owner, when the file owner uploads a file to the cloud server, they are essentially placing the file into the cloud. This upload is intended for the file to be accessible from anywhere via the cloud server. The file owner generates two keys for the file's security; one key is concealed behind an image, while the other key is held by the file owner. 4.2.2.

User

In this context, a user refers to the individual accessing the file. When a user wishes to retrieve a file from the cloud server, they must first obtain two keys. Therefore, the user sends a request to the file owner to access the first key. Once the user receives the first key from the file owner, the image key will then be displayed on the user's side. Finally, the user inputs both keys into the cloud server, allowing access to the file from the cloud server.

Cloud server

We facilitate both the upload and download processes exclusively through the cloud server. The original file content is stored solely on this cloud server. The cloud server can connect to all users and file owners due to its open nature. When users wish to download files, they only need to search the cloud server. When the user enters the correct keys, the file is downloaded from the cloud server.

Database Design

A database is an organized collection of related data that is stored with minimal redundancy, allowing for quick and efficient access by multiple users. The primary goals of database design are to ensure that data access is straightforward, cost-effective, and adaptable for users. Data within the system must be stored and retrieved from the database.

The process of designing the database is a component of system design. During the analysis phase, data elements and structures to be stored have been identified, and these are organized and assembled to create the data storage and retrieval system.

Table 3.1: Owner Registration

Fields	Type	Null	Constraints	Description
Firstname	varchar(45)	No		Owner First Name
Last name	varchar(45)	No		Owner LastName
User name	varchar(45)	No		Owner User Name
Mail	varchar(45)	No		Owner Email
Mobile	varchar(45)	No		Owner Mobile
Date of birth	varchar(45)	No		Owner Date of birth
City	varchar(45)	No		Owner City name
Address	varchar(45)	No		Owner Address
Password	varchar(45)	No		Owner Password
Confirm password	varchar(45)	No		Owner Confirm Password
Gender	varchar(45)	No		Owner Gender

Table 3.2: Provider Register

Fields	Type	Null	Constraints	Description
Firstname	varchar(45)	No		Provider First Name
Last name	varchar(45)	No		Provider LastName
User name	varchar(45)	No		Provider User Name
Mail	varchar(45)	No		Provider Email
Mobile	varchar(45)	No		Provider Mobile
Date of birth	varchar(45)	No		Provider Date of birth
City	varchar(45)	No		Provider City name
Address	varchar(45)	No		Provider Address
Password	varchar(45)	No		Provider Password
Confirm password	varchar(45)	No		Provider confirm Password
Gender	varchar(45)	No		Provider Gender

Table 4.3: File Upload

Fields	Type	Null	Constraints	Description
User name	varchar(45)	No		Owner User Name
Date	varchar(45)	No		File upload date
key1	varchar(45)	No	Primary Key	Key of the upload file
file12	LONGBLOB	No		Upload File
encpwd	varchar(45)	No		Password of the file
filesize	varchar(45)	No		Size of the file
filename	varchar(45)	No		Name of the file
status	varchar(45)	No		Status of the file

Input Design

The input of a system refers to the information provided to it. This information is utilized for subsequent processing to derive meaningful insights that aid in decision-making. Input design involves transforming user-oriented inputs into a format suitable for computer systems.

Input is a crucial component of the overall system design, necessitating careful consideration. Inaccurate input data frequently lead to errors during processing. Effective input design can mitigate user-entered errors. The data entered must be verified for accuracy and the nature of any errors. Suitable error messages should be presented. If invalid data is inputted, the user must be prevented from submitting that data.

Output Design

The output generated by a computer serves as the primary and most direct source of information for the user. A well-designed and comprehensible output enhances the system's interaction with the user and supports decision-making. The output design was actively examined during the study phase. The goal of output design is to define the content and format of all documents and reports in a manner that is both appealing and functional.

CONCLUSION AND FUTURE ENHANCEMENT

In the current system, file owners upload their files to a cloud server. Consequently, multiple file owners have access permissions on the same cloud server, which allows one owner to potentially access another owner's files. This situation increases the risk of misuse of files by other owners. To address this issue, we are implementing an encryption technique that involves the generation of two types of keys. When a file owner wishes to access their file from the cloud server, both keys are required for security purposes. However, there is also a risk that hackers may attempt to compromise these keys. Therefore, in our proposed system, we are generating two keys simultaneously, with one key concealed behind an image.

When a file owner needs to access their file from the cloud server, they must first enter one key, after which the image key will be revealed. This method is primarily designed to prevent hackers from accessing the keys.

Our IEEE concept aims to maintain information across a vast number of cloud computing environments by incorporating additional features, as outlined below: 1. A future enhancement of this project involves storing files in a private cloud storage system. 2. Files will be transferred to facilitate interaction between private cloud users and data owners, ensuring that unauthorized users cannot access files from the cloud server. 3. All these processes will be communicated to our project users worldwide through this application.

REFERENCES

[1]. Muqadar Ali et.al "Efficient Secure Privacy Preserving Multi Keywords Rank Search over Encrypted Data in Cloud Computing" Journal of Information Security and Applications ,Volume 75, June 2023, 103500.
[2]. Cong Wang et al., "Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data", IEEE Transactions on parallel and distributed systems, vol. 23, no. 8, August 2012
[3]. Wenhai Sun et al., "Privacy-Preserving Multikeyword Text Search in the Cloud Supporting Similarity-based Ranking", the 8th ACM Symposium on Information, Computer and Communications Security, Hangzhou, China, May 2013.
[4]. Jiadi Yu, Peng Lu, Yanmin Zhu, Guangtao Xue, Member, IEEE Computer Society, and Minglu Li, "Toward Secure Multikeyword Top k Retrieval over Encrypted Cloud Data", IEEE Journal of Theoretical and Applied Information Technology 10th August 2014. Vol. 66 No.1 © 2005 - 2014 JATIT & LLS. All rights reserved. ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195 64 Transactions on dependable and secure computing, vol. 10, no. 4, July/August 2013
[5]. Ning Cao et al.," Privacy-Preserving MultiKeyword Ranked Search over Encrypted Cloud Data", IEEE Transactions on parallel and distributed systems, vol. 25, no. 1, jan 2014
[6]. Wenhai Sun et al., "Protecting Your Right: Attributebased Keyword Search with Finegrained Ownerenforced Search Authorization in the Cloud", IEEE INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014
[7]. M. Kuzu, M. S. Islam, and M. Kantarcioglu, ''Efficient similarity search over encrypted data,'' in Proc. IEEE 28th Int. Conf. Data Eng., Apr. 2012, pp. 1156–1167.
[8]. Beimel, ''Secure schemes for secret sharing and key distribution,'' Ph.D. dissertation, Technion-Israel Inst. Technol., Haifa, Israel, 1996.
[9]. Sahai and B. Waters, ''Fuzzy identity-based encryption,'' in Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn., in Lecture Notes in Computer Science, vol. 3494, 2005, pp. 457–473.
[10]. V. Goyal, O. Pandey, A. Sahai, and B. Waters, ''Attribute-based encryption for fine-grained access control of encrypted data,'' in Proc. 13th ACM Conf. Comput. Commun. Secur., 2006, pp. 89–98.
[11]. Beimel and A. Ben-Efraim, ''Multi-linear secret-sharing schemes,'' in Proc. Theory Cryptogr. Conf., in Lecture Notes in Computer Science, vol. 8349, 2014, pp. 394–418.
[12]. D. Boneh and M. Franklin, ''Identity-based encryption from the Weil pairing,'' in Advances in Cryptology (Lecture Notes in Computer Science), vol. 2139, J. Kilian, ed. Berlin, Germany: Springer-Verlag, 2001, pp. 213–229.