# PRESERVING DATA USING MULTI-KEYWORD RANKED SEARCH OVER ENCRYPTED CLOUD STORAGE

**[1]M.Prabu, [2]E.Ganesh**

[1]Assistant Professor, [2]Research Scholar,
[1] Department of ECE
[1] Misrimal Navajee Munoth Jain Engineering College, Chennai -97, India.

***Abstract***: As cloud computing gains popularity, more data owners are opting to move their complex data management systems from local sites to commercial public clouds for the benefits of flexibility and cost savings. However, to ensure data privacy, sensitive information must be encrypted before outsourcing, rendering traditional data utilization through plaintext keyword search ineffective. Therefore, there is a growing need for encrypted cloud data search services. With the vast number of data users and documents in the cloud, these services must allow for multi-keyword queries and provide similarity rankings to facilitate effective data retrieval. While previous research on searchable encryption has mostly focused on single keyword or Boolean keyword searches, this paper introduces the challenge of privacy-preserving Multi-keyword Ranked Search over Encrypted cloud data (MRSE) and sets strict privacy requirements for such a secure cloud data utilization system. We adopt the efficient principle of "coordinate matching" among the various multi-keyword semantics to capture the similarity between the search query and data documents. We also utilize "inner product similarity" to quantify this principle for similarity measurement. Our proposed scheme includes a basic MRSE that employs secure inner product computation and a significantly improved version that meets different privacy requirements in two levels of threat models. We thoroughly analyse the privacy and efficiency guarantees of these proposed schemes, and experiments on a real-world dataset confirm that our proposed schemes introduce low overhead on computation and communication.

***Index Terms-*** **Data privacy, MRSE, Privacy, efficiency.**

## I. INTRODUCTION

Cloud computing has long been envisioned as computing-as-a-utility, where cloud customers can remotely store their data in a shared pool of configurable computing resources and enjoy on-demand, high-quality applications and services. The great flexibility and economic savings offered by the cloud are prompting individuals and enterprises to outsource their local complex data management systems to commercial public clouds. However, to protect data privacy and prevent unauthorized access in the cloud and beyond, data owners may need to encrypt sensitive data such as e-mails, personal health records, photo albums, tax documents, financial transactions, and more before outsourcing it to the cloud. This renders traditional data utilization services based on plaintext keyword search obsolete. The impractical solution of downloading all the data and decrypting it locally due to the high bandwidth costs associated with cloud-scale systems. Additionally, storing data in the cloud serves no purpose if it cannot be easily searched and utilized. Thus, it is crucial to explore privacy-preserving and effective search services over encrypted cloud data. This problem is particularly challenging due to the potentially large number of on-demand data users and the vast amount of outsourced data documents in the cloud, making it extremely difficult to meet the requirements of performance, system usability, and scalability.

To achieve effective data retrieval, a ranked search system is necessary to sort the large number of documents instead of providing undifferentiated results. This system allows data users to quickly find the most relevant information, reducing the burden of sorting through every match in the content collection. Additionally, this system eliminates unnecessary network traffic by sending back only the most relevant data, which is desirable in the “pay-as-you-use” cloud paradigm. However, privacy protection must also be considered, and the ranking operation should not leak any keyword-related information. To enhance the search result accuracy and user experience, the ranking system must support multiple keyword searches. This is a common practice among web search engines, where users provide a set of keywords to retrieve the most relevant data. “Coordinate matching” is an efficient similarity measure among such multi-keyword semantics and has been widely used in plaintext information retrieval. However, applying this measure in an encrypted cloud data search system poses a significant challenge due to strict requirements such as data privacy, index privacy, keyword privacy, and others.

In literature, searchable encryption is a technique that treats encrypted data as documents and allows secure searching through a single keyword to retrieve relevant documents. However, directly applying these approaches to a secure large-scale cloud data utilization system may not be suitable as they are developed as cryptographic primitives and may not meet the high service-level requirements such as system usability, user search experience, and information discovery. While recent designs have attempted to support Boolean keyword search to enrich search flexibility, they are still inadequate in providing acceptable result ranking functionality for users. Various works have addressed the challenge of providing secure ranked search over encrypted data, but only for single keyword queries. However, the problem of designing an efficient encrypted data search mechanism that supports multi-keyword semantics without compromising privacy remains an open challenge. In this paper, we introduce a solution to this problem by defining and addressing the issue of multi-keyword ranked search over encrypted cloud data (MRSE) while ensuring strict system-wide privacy in the cloud computing paradigm. To capture the relevance of data documents to the search query, we select the efficient similarity measure of "coordinate matching," which aims to maximize the number of matches between query keywords and document keywords. Specifically, we use "inner product similarity" to quantitatively evaluate the similarity measure of a document to the search query, based on the number of query keywords that appear in the document. This approach enables us to provide a secure and efficient multi-keyword ranked search over encrypted cloud data.

During the index construction, a binary vector is associated with each document as a sub-index. Each bit in the vector represents whether the corresponding keyword is present in the document. Similarly, the search query is also represented as a binary vector, where each bit indicates whether the corresponding keyword appears in the query. The similarity between the query and the data can be measured using the inner product of their respective vectors. However, directly outsourcing the data or query vectors can compromise the index or search privacy. To address this challenge, we propose a basic idea for MRSE using secure inner product computation, adapted from a secure k-nearest neighbour (KNN) technique. We then present two significantly improved MRSE schemes, developed step-by-step, to meet various privacy requirements in two threat models, with increased attack capabilities.

## II. RELATED WORKS

To facilitate data sharing in the cloud, a combination of public key cryptosystems technique and identity-based encryption was utilized [1]. The objective of the Public Key Exchange scheme that is Addressable (PKA) encryption scheme is to serve as a supplement or replacement for current public key infrastructures. Presented the PKA, a public key scheme based on addressable cryptographic tables [2]. Proposed plan of action aims to address concerns regarding data privacy in the cloud by leveraging cryptographic algorithms to improve security from various perspectives of cloud customers [3]. Two innovative solutions for Authorized Private Keyword Search (APKS) have been proposed utilizing the cryptographic primitive known as Hierarchical Predicate Encryption (HPE) [4]. Secure file sharing mechanism for cloud computing was explored, utilizing the disintegration protocol (DIP). Additionally, a novel contribution of a seamless file sharing technique was introduced, which allows for secure file sharing among different clouds without the need to share an encryption key [5]. A symmetric encryption technique has been proposed, where the algorithm operates on the ASCII code of each value within the original data. The encryption process occurs prior to transmitting the data to the cloud [6]. The problem of securely searching for ranked keywords over encrypted cloud data has been defined and solved. The implementation of ranked search significantly improved the system's usability by allowing for search result relevance ranking, as opposed to providing undifferentiated results. This approach also enhances file retrieval accuracy [7].

## III. SYSTEM ANALYSIS

System analysis is a problem-solving technique that involves breaking down a system into its component parts to study how they interact to achieve their goals. By gathering and interpreting data, system analysis diagnoses problems and recommends solutions to improve the system. In the cloud, where there are large numbers of data users and documents, effective data retrieval requires search services that allow multi-keyword queries and provide similarity rankings. However, current searchable encryption techniques are limited to single keyword or Boolean searches that do not differentiate search results. To address this, we propose a privacy-preserving multi-keyword ranked search over encrypted cloud data (MRSE) and establish a set of strict privacy requirements. We use the efficient principle of “coordinate matching” and develop a tree-based index structure that incorporates a “greedy depth first search” algorithm to enable MRSE.

## IV. TECHNOLOGY USED

Typically, programming languages require either compilation or interpretation to run a program on a computer. However, the Java programming language is unique in that it uses both. When compiling a Java program, it is translated into an intermediate language called Java bytecodes. These bytecodes are platform-independent codes that can be interpreted by the interpreter on the Java platform. The interpreter processes each bytecode instruction and executes them on the computer. While compilation only happens once, interpretation takes place each time the program runs. Additionally, Java bytecodes can be used as machine code instructions for the Java Virtual Machine (Java VM), which is implemented by every Java interpreter, including those found in web browsers and development tools. This feature of Java bytecodes enables the "write once, run anywhere" capability, as a program written in the Java programming language can be compiled into bytecodes on any platform with a Java compiler, and then run on any implementation of the Java VM on various operating systems such as Windows 2000, Solaris, or iMac.

**Java Platform:** A program's platform refers to the environment, either hardware or software, on which it runs. Common platforms include Windows 2000, Linux, Solaris, and MacOS, which are typically a combination of the operating system and hardware. However, the Java platform is unique in that it is a software-only platform that operates on top of other hardware-based platforms. The Java API includes a vast array of pre-made software components that offer various capabilities, such as graphical user interface (GUI) widgets. These components are organized into packages, which are libraries comprising related classes and interfaces.

**Java SDK:** The Java programming language is a powerful, yet easy-to-learn object-oriented language, particularly for programmers familiar with C or C++. Program metrics comparisons suggest that a program written in Java can be four times smaller than the same program in C++. The language promotes good coding practices and includes garbage collection to avoid memory leaks. Its object orientation, JavaBeans component architecture, and wide-ranging, easily extendible API enable the reuse of tested code and reduce the introduction of bugs. To ensure program portability, it is recommended to avoid using libraries written in other languages. Applets can be easily upgraded from a central server, taking advantage of the feature of allowing new classes to be loaded "on the fly," without having to recompile the entire program. Figure.1 shows the JAVA IDE.

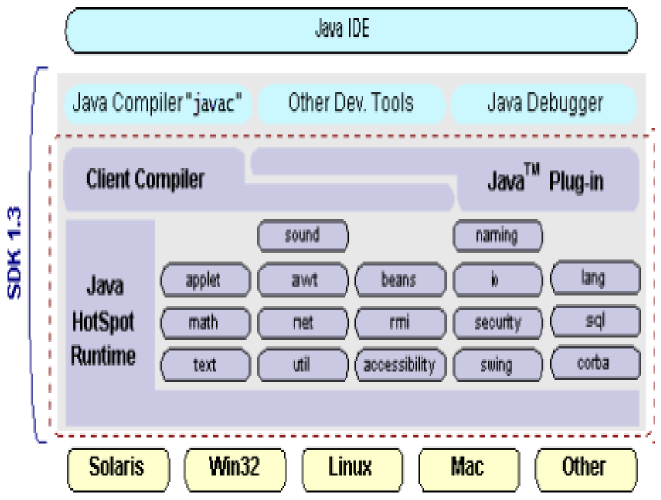

**Figure.1** JAVA IDE

**ODBC:** Microsoft Open Database Connectivity (ODBC) serves as a standard programming interface for both application developers and database system providers. Prior to ODBC's adoption as a de facto standard for Windows programs to interact with database systems, programmers had to use proprietary languages tailored to each database system they wanted to connect with. With ODBC's introduction, the choice of database system becomes almost irrelevant from a coding perspective, freeing developers to focus on more pressing concerns than syntax needed to port their program when business needs change. Using the ODBC Administrator located in Control Panel, developers can specify the database associated with a data source that an ODBC application program uses. Think of an ODBC data source as a door labelled with a name, leading to a specific database. The benefits of this approach are numerous, leaving one wondering if there's a catch. The only drawback is that ODBC isn't as efficient as communicating directly with the native database interface. Some have claimed that ODBC is too slow.

**JDBC:** Sun Microsystems developed Java Database Connectivity, or JDBC, as an attempt to establish an independent database standard API for Java. JDBC provides a generic SQL database access mechanism that offers a consistent interface to a range of RDBMS. This consistency is achieved by utilizing "plug in" database connectivity modules, or drivers. Database vendors who wish to have JDBC support must provide the driver for each platform that the database and Java run on. To encourage wider adoption of JDBC, Sun modelled its framework on ODBC. As noted earlier in this chapter, ODBC is widely supported on various platforms. By basing JDBC on ODBC, vendors can bring JDBC drivers to market more quickly than developing a completely new connectivity solution.

## V. SYSTEM IMPLEMENTATION

System design refers to the process of defining the architecture, components, modules, interfaces, and data of a system to meet specific requirements. It can be viewed as the application of systems theory to product development and shares similarities with systems analysis, system architecture, and systems engineering. Unified Modelling Language (UML) is a graphical visualization language that consists of symbols and connectors used to create process diagrams, often used to model computer programs and workflows. UML diagrams can also be used to visualize website structures and user interactions. Use case diagrams, a type of behavioural diagram, are created from use-case analysis and provide a graphical overview of the functionality provided by a system in terms of actors, their goals, and any dependencies between use cases. On the other hand, a data-flow diagram illustrates the flow of data through a system or process, providing information about the inputs and outputs of each entity and the process itself. Unlike flowcharts, data-flow diagrams have no control flow, decision rules, or loops, but specific operations based on the data can still be represented. System Architecture is shown in Figure.2.

**Admin Module:** This module facilitates secure file viewing and uploading for the server. The administrator logs in using a key and logs out by changing the key. After logging in, the administrator can change their password and access details on user downloads and file requests through a flowchart. Additionally, the administrator can activate new user accounts.

**Client Module:** This module facilitates file searching for clients by using multiple keywords and provides an accurate result list based on the user's query. After selecting the desired file, the user can register their details and request access to download the file or data. If the data owner approves the request, they will provide key access to download the requested file or data. If the owner declines the request, the encrypted stored data will not be terminated, and the requested user will not have access to the data.

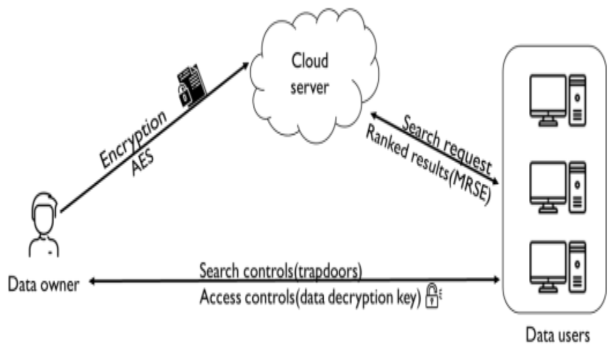

**Figure.2** System Architecture

**Encrypt Module:** This module is designed to assist the server in encrypting documents using the AES algorithm. The encrypted data is stored with a randomly generated key, making the data unreadable without the corresponding key. The Java key generator is used to generate these random keys.

**Multi-Keyword Module:** The purpose of this module is to assist the user in obtaining accurate search results based on multiple keywords. The user can enter a query consisting of multiple words, which the server will then split into individual words and search for in the database. The server will display a list of matching words from the database, and the user can select the desired file from that list.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we introduce and address the issue of multi-keyword ranked search over encrypted cloud data, and establish a range of privacy requirements. From various multi-keyword semantics, we choose the efficient similarity measure of "coordinate matching" to capture the relevance of outsourced documents to the query keywords, using "inner product similarity" to quantitatively evaluate such similarity measure. To support multi-keyword semantics without compromising privacy, we propose a basic idea of MRSE using secure inner product computation. We present two enhanced MRSE schemes to achieve various stringent privacy requirements in two different threat models. We also examine further improvements of our ranked search mechanism, including supporting more search semantics and dynamic data operations. We offer a thorough analysis investigating the privacy and efficiency guarantees of the proposed schemes, and experiments on the real-world data set demonstrate that our proposed schemes introduce low overhead on both computation and communication. In future work, we will explore verifying the integrity of the rank order in the search result with the assumption that the cloud server is trusted, and we will add more formats that can be uploaded to our software platform.

## REFERENCES

**[1]** Adeppa, S. 2015. Data Sharing in Cloud Storage using Identity Encryption Technique. International Journal on Emerging Technologies, 6(1), 115.

**[2]** Habib, B., Cambou, B., Booher, D and Philabaum, C. 2017. Public key exchange scheme that is addressable (PKA). IEEE Conference on Communications and Network Security (CNS), Las Vegas, NV, USA. 392-393.

**[3]** Khan, S. S., and Tuteja, R. R. 2015. Security in cloud computing using cryptographic algorithms. International Journal of Innovative Research in Computer and Communication Engineering. 3(1): 148-155.

**[4]** Li, M., Yu, S., Cao, N and Lou, W. 2011. Authorized Private Keyword Search over Encrypted Data in Cloud Computing. 31st International Conference on Distributed Computing Systems, Minneapolis, MN, USA. 383-392.

**[5]** Rawal, B.S. and Vivek, S. S. 2017. Secure Cloud Storage and File Sharing. IEEE International Conference on Smart Cloud (Smart Cloud), New York, NY, USA. 78-83.

**[6]** Sugumar, R., & Imam, S. B. S. 2015. Symmetric encryption algorithm to secure outsourced data in public cloud storage. Indian journal of science and technology, 8(23):1.

**[7]** Wang, C., Cao, N., Ren, K and Lou, W. 2012. Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data. IEEE Trans. Parallel and Distributed Systems, 23(8): 1467- 1479.