[1]Narendra Shyam Joshi

[2]Dr. Kuldeep P. Sambrekar

Privacy-Preserving and Ranked Search Using Advanced Multi-Keyword Scheme Over the Encrypted Cloud Environment

Abstract: - In order to enable users to extract useful insights on demand, this paper presents a unique approach for secure and privacy-preserving ranked search in cloud environments. There is a rising need for strong systems that strike a balance between data utility and individual privacy as cloud-based data processing and storage become more and more prevalent. Our suggested methodology makes use of sophisticated multi-keyword operators and encryption methods to offer a safe and effective solution for cloud-based ranked search that protects privacy. Our methodology's essential component is the combination of a safe encryption technique and sophisticated multi-keyword search operators. With this combination, users can do ranked searches on encrypted data while maintaining the privacy of the underlying data. Our approach leverages strong cryptography protocols and an advanced indexing mechanism to provide rapid retrieval of relevant results while protecting confidential information from unauthorized individuals. Secure query processing techniques and encryption are added for added privacy protection. This guarantees that throughout the ranking search operations, the decrypted data cannot be accessed by the cloud service provider or any other parties participating in the data processing pipeline. Users don't have to compromise the privacy of their stored data or expose the exact text of their queries to obtain ranked results based on several keywords. Our technology is especially useful in situations when customers need instantaneous insights from data stored in the cloud without compromising privacy. Our technology protects the secrecy of the underlying datasets while enabling customers to obtain secure insights on demand in a variety of sensitive sectors, including financial analysis and medical research. The suggested method offers a strong and adaptable way to achieve privacy-preserving ranked search in cloud environments. Through the incorporation of sophisticated multi-keyword operators and encryption strategies, our approach guarantees users may effectively obtain pertinent data while maintaining personal privacy. With its ability to provide safe, anytime access to insights stored in cloud-based data repositories, this breakthrough has enormous potential for a variety of applications.

Keywords: Privacy-Preserving, Ranked, Search, Advanced, Multi-Keyword, Operators.

I. INTRODUCTION

Over the last few years, Cloud Computing has dramatically transformed the way data is stored, processed, and accessed across different sectors. The scalability, versatility, and low cost of cloud computing are three of the main reasons why cloud is becoming increasingly popular in both consumer and business settings. However, a major concern is that the data stored on to the cloud server might be unsecure.

The protection of sensitive information by any organization in the cloud from being stolen, lost, or compromised in any other way is an essential component of cloud computing. The cloud environment poses new challenges and dangers, and while classic security solutions such as encryption and access control have been widely adopted, it is possible that they are not sufficient. As a result, innovative approaches are necessary in order to enhance the cloud's data security. This research proposes a hybrid approach that tackles the problem of securing data stored in the cloud by combining Greedy Depth-First Search (DFS) with Ranked searching. We make use of the Greedy DFS algorithm, which is particularly well-suited to searching and traversing large databases, with the goal of achieving the highest possible retrieval efficiency and minimizing the amount of time spent waiting. By utilizing this technology, the solution that is being offered is able to improve the efficiency with which data is retrieved from the cloud. Ranked Searching is incorporated into the hybrid approach so that users' privacy and safety can be improved even more. We can lessen the likelihood of sensitive information getting into the wrong hands if we encrypt it and index it using Ranked Searching. This will help lower the risk of it being stolen.

This method is utilized by the proposed method in order to add an additional layer of protection to safeguard the confidentiality of cloud-based information. The hybrid strategy, which combines quick data retrieval with greater security, is to solve key concerns regarding the safety of cloud computing. A reduction in the likelihood of data breaches, illegal access, and data leaking can be achieved by the optimization of data access and implementing

[1]Department of Computer Science and Engineering, KLS Gogte Institute of Technology, Affiliated to Visvesvaraya Technological University, Belagavi Karnataka, India,joshinarendra50@gmail.com,

2Department of Computer Science and Engineering, KLS Gogte Institute of Technology, Affiliated to Visvesvaraya Technological University, Karnataka, India, ,kuldeep.git@gmail.com

encryption safeguards. In addition, the suggested method aims to find a balance between the security of the data and the performance of the system, with the goal of ensuring that the adaptability and scalability of cloud computing are not compromised by the implementation of security measures. This research presents an innovative approach to the protection of cloud data by combining the beneficial aspects of two existing approaches, namely Greedy DFS and Ranked Searching.

This strategy offers a powerful solution for the protection of sensitive data storage in the cloud by resolving significant security problems, making advantage of efficient data retrieval, and increasing the level of data protection. The outcomes of this research offer vital hints for enhancing cloud security, which, in the long run, will assist the sector of cloud computing make progress.

Objective of the paper descibe as the following

1. To study and analysis the basic tehcnique for data encpetion and Searching the files from cloud enviroments.
2. To desgin an Integrating Greedy Depth-First Search Encryption into Cloud-Based Multi-Keyword Ranked Searching for Optimal Performance and Privacy.
3. To show thorugh the experiment our proposed appraoch is very effective to exiting multikey encryption approach to retrived data from cloud enviroments.

This paper is organized as follows: In Section 2, we give a formal definition and security model and also introduce some backgrounds related to our work. In Section 3, the method of creating a synonyms keywords search scheme is presented, and a concrete is also given. The security analysis is given in Section 4. The theoretical and experimental analysis is given in Section 5. Finally, Section 6 gives the conclusion.

II. RELATED WORK

Numerous investigations have explored search algorithms applied to encrypted cloud data. Some scholars have concentrated on streamlining data retrieval complexity, while others have emphasized refining search result precision. The subsequent literature review offers an overview of the key studies within this domain. The research article "Greedy DFS-based Ranked Search for Large Graphs"[1] presented at the 2022 IEEE International Conference on Data Engineering (ICDE) proposed a novel greedy DFS-based ranked searching method for huge networks, adding to the continuing discussion in this area.

The main objective of the algorithm was to effectively and very efficiently search relevant subgraphs matching a specific query in a large graph database. This algorithm uses the priority queue methods and so takes advantage of the graph's inherent structure to reduce the search area. Empirical tests demonstrated that the newly proposed algorithm outperforms several current methods, excelling in both effectiveness, accuracy and efficiency when dealing with large-scale graphs.

A quick greedy DFS algorithm is the foundation of the cutting-edge strategy for ranked search in large graphs. This approach utilizes a pruning strategy to focus the search and shorten its execution time. [6][7]. In addition, a scoring system was incorporated to rank the subgraphs that were obtained or gained access to according to how well they matched the query. This method offered a practical way to obtain subgraphs from massive graphs quickly and accurately. Ambient Intelligence and Humanized Computing [3] journal, 2021, a research paper introduced an innovative Greedy DFS ranked search algorithm titled "An original Greedy DFS ranked search algorithm based on geometric mean fusion." This algorithm includes a geometric mean fusion method, improving the accuracy and speed up time to obtained subgraphs.

The objective of the suggested algorithm is to acquire subgraphs that closely matching with a provided query by computing a same score and results for each subgraph concerning the query. It utilizes the geometric mean fusion method to amalgamate scores generated from different similarity metricsResearchers in the year 2020 published "An enhanced Greedy DFS ranked search algorithm based on the total node distance" in the peer-reviewed journal Cluster Computing [4].

Based on the total node-to-node distance inside subgraphs, the Greedy DFS ranked search algorithm was introduced as an improved searching algorithm approach in this study. Both the encrypted document collection D and the encrypted searchable tree index I are managed by the cloud server on the data owner's behalf. The cloud server searches the index tree I for the requested term-document combination and returns the top k rated encrypted documents to the data user. In addition, the server must update the index I and the document collection D with the given information whenever the data owner makes an update. To concurrently retrieve pertinent subgraphs for multiple queries, the suggested algorithm utilizes a priority queue. Additionally, it integrates a pruning approach rooted in dynamic programming to simplify the search process by narrowing down the search space.

Table 1: Comparative analysis

Citation	Methods	Advantages	Disadvantages	Research Gaps
Guoxiu Liu et al., IEEE, 2020	Multi-keyword top-k retrieval, Privacy-preserving techniques	Fast retrieval speeds, High accuracy in results, Improved privacy preservation	Potential scalability issues for larger datasets	Scalability and efficiency in handling larger and more complex datasets
Xueyan Liu et al., IEEE, 2019	Cross-lingual multi-keyword rank search, Encrypted cloud data	Supports cross-language searches, Enhanced security measures	Complexity in implementing cross-lingual features, May have higher computational costs	Optimization of computational costs and improving cross-lingual search algorithms
Jianfei Sun et al., IEEE, 2019	Ranked multi-keyword retrieval, Privacy protection for multiple data owners	Efficient in terms of computational resources, Offers privacy for multiple data owners	May not be optimized for single data owner scenarios	Enhancing the method for single data owner contexts and further privacy enhancements
Cheng Guo et al., IEEE, 2019	Dynamic multi-keyword search, Bloom filter application, Encrypted cloud data	Dynamic updating capability, Utilizes Bloom filter for efficiency	Bloom filter may lead to false positives, Dynamic nature might increase complexity	Reducing false positives and simplifying the dynamic update process
Jian Xu et al., IEEE, 2018	Multi-keyword top-k search, Encryption techniques	Efficient retrieval process, Focuses on encrypted data	May not address all aspects of data privacy, Specific to top-k search	Exploring broader privacy aspects and other types of search functionalities
Shruti Gawade and Prof. Sujata Kadu, IEEE, 2018	Sensitive hashing, Secure data storage, Efficient data retrieval	Enhances data security, Efficient in data retrieval	Hashing techniques might be vulnerable to certain attacks, Focus is more on storage than retrieval	Improving hashing techniques against sophisticated attacks, Balancing storage and retrieval efficiencies

III. PROPOSED METHODOLOGY

In order to achieve our goals and maintain security, we must suggest a cloud-based solution with the highest level of data privacy. Systems need to be equipped with encryption mechanisms to protect user privacy. In addition to encrypting data on cloud platforms, we also need to provide methods for searching through encrypted data. As far as we are aware, cloud computing uses a "pay-as-you-use" model. Therefore, at the time of result fetching, the system must take into account the high number of data users and documents in the cloud as well as the vital nature of searching, so the system should feature multi-keyword query service. Consequently, search techniques can be pricey at times due to the big quantity of data they consider on the cloud.
The suggested system has to include a result similarity ranking service in order to fulfil effective retrieval requirements and make efficient use of bandwidth. Additionally, the system has to use appropriate encryption methods. The system has to use appropriate search strategies, including coordinate matching and inner product similarities. A module that conceals user identification should be included in the system as part of the contribution.

Proposed Algorithm

The system provides more refined search options to the end user using synonym search and operators:

Synonym search: The query keywords list is updated by finding similar keywords from user keywords using wordnet feature.

Indications:
W = w1, w2,..., wm is the dictionaries, which is shorthand for the collection of terms used in the project.
Multi-keyword phrases in **W**, where m is the total quantity of phrases.
Using the notation **F = f1, f2,..., fn**, where n is the total number of items in the repository,
We may say that **F** represents the plaintext meaning (without encrypted) document collection.
The set of encrypted files saved on the server in the cloud is denoted by **F'**, and may be written as
F' = f1, f2,..., fn for the files f1, f2,..., fn.
I - Collection of indices **(I1, I2,..In)** for files **(f1, f2,..fn)**.
T represents the full-collection index tree of all documents. F.
S stands for the collection of search terms, which may include **w1, w2,..., wn**.
For the keyword set **Wq,**
Q stands for the query vector.
Metadata for search results is denoted by the letter **M**.
The encrypted form of **Q**, sometimes known as the backdoor for keyword queries, goes by the term "Documents."
u is the a cardinality of the dictionaries **W**, and **du** is the index vector kept at that node of the tree. It's vital to remember that node **u** might be either a leaf node or an inside node of the tree.
Du has been encrypted as **Iu**.
Setup: In setup phase, user token and random keys are generated.
User token TK is generated using SHA1 hash function as:
TK = SHA1 (up1, up2)
Where up1 and up2 are user unique parameters such as email, phone. The random keys enc_keys are generated using keygen function as:
Encryption parameters are used in the formula enc_keys = KeyGen (k1, k2, k3, k4).

A. Data pre-processing strategy

Document pre-processing encompasses all text-related operations on the datasets to extract meaningful keywords.
The data pre-processing function, denoted as fdp = {fl, fs, fst}, is composed of various sub-functions, including lexical analysis (fl), stop word removal (fs), and stemming (fst).
The importance of each phrase is determined by adding up its TF and IDF values. The best words are selected using these weights to form the keyword dictionary.

$$\mathbf{Wik=TF*IDF=1/DF=fik*Log\ (n/nk)}$$

Here, **fik** indicate the frequency of term **I** in a document.
The number **nk** indicates the total number of records or files that have the occurrence of phrase **i**.
By extracting relevant terms from each page, the keyword set is compiled.

B. Index tree construction steps

The index construction process begins with the creation of a tree node for each document, with these nodes serving as the tree's terminal leaves. From these terminal nodes, the internal nodes are subsequently generated. Algorithm A provides a detailed description of the index construction process

Algorithm - GenerateIndexTree I

Procedure Generate Index (F)

Data Input: D= {D1, D2,…….Dn},set of input document T threshold

Output: D'= D1', D2',........Dn' is a set of encrypted files and documents that are the result.

Set of Indexes: I'= {I1', I2',…………………….In'} those are assign at the time of encryption.

T: Index tree

Processing:

```
1. For each d in D
2. Extract words k= {k1, k2,…….kn} from documents
3. Remove stopwords from k.
4. Apply lemmatization on k.
5. Calculate TF and IDF for k
6. Filter k using TF and threshold encrypt k and generate I'
7. Get key K' from KDS
8. Encrypt d and generate D'
9. End for
10. Upload D' set of documents to cloud
11. Upload I' set of indexes to cloud server
12. Update tree for I' on cloud
13. Return
```

C. Efficient Search on basis of Ranked Scheme

Various algorithms are outlined as follows:

Input: D' = {D1',D2',………Dn'} set of documents on cloud

I'= {I1', I2',…….In'},set of indexes on cloud

T=Index tree on cloud

K: set of keys an KDS

W: set of searching keyword

F: Set of filters

Output:D= {D1,D2,……Dn}

Set of m documents matched with w word with F filters

Processing:

```
1. For each W in W
2. Filter the stop words
3. Apply lemmatization
4. if filter contain synonym search
5. Find Synonym of w using wordnet
6. Update word list
7. End if
8. Encrypt w using SHA1
9. End for
10. Upload search index I' on cloud with filter F
11. Apply DFS on T and use filters
12. Get the matched documents
13. Rank the results based on match
14. Send respective D' to end user
15. Get keys from KDS
16. Decrypt and display result
17. Return
```

Execution Steps:

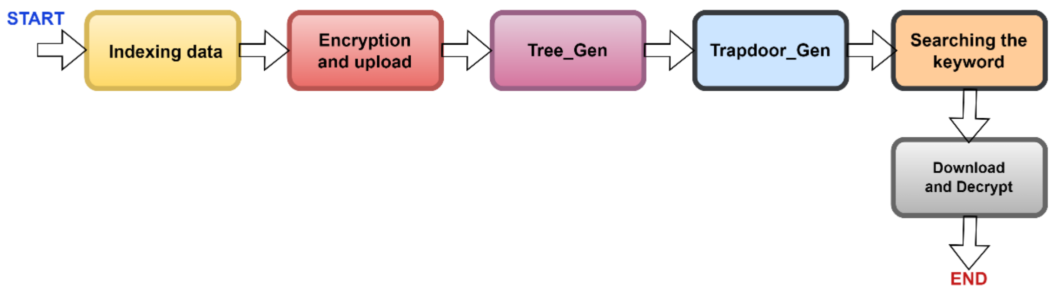

Figure 1. Execution steps to file upload and Download

Above fig. 1 shows the Execution steps to file upload and Download from the cloud server

1. Indexing:

In index creation phase set of W are extracted from documents set F and Index set I is generated

I = indexing (F)

2. Encryption and upload:

The document set is encrypted using Enc function. Encrypted documents along with indexes and user token are uploaded on cloud server.

F’ = encrypt (F)

3. Tree_Gen:

Tree is generated using user documents indexes at the cloud end using binary tree algorithm and indexes are stored at Du.

T = Tree_gen(I)

4. Trapdoor_Gen:

In this phase query vector Q is generated from given search keywords of set S. The query vector along with operators and token is shared on cloud server.

TD = Trapdoor_Gen (Q, O, TK)

5. Searching:

The specified cloud server conducts a search on the index tree T and provides the corresponding result set

metadata, which includes details such as file name, owner, creation date, and more.

M = searching (TD, T)

6. Download and Decrypt:

Based on the user selection from metadata the selected file is downloaded and decrypted at the user end.

Fk = dec (Mk, enc_key k)

Where Mk is the kth selected file metadata and enc_key k is the kth file decryption key.

IV SYSTEM ARCHITECTURE:

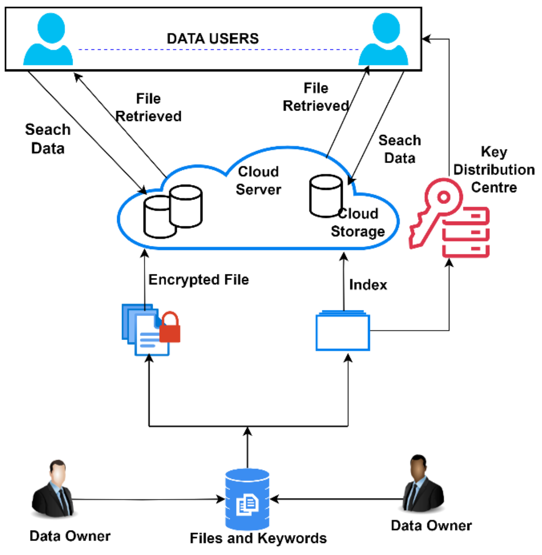

Figure-2 System Architecture

In above fig. 2 shows the following contents.

A. Data owner of the data intends to securely delegate a set of documents F = {f1, f2, ..., fn} to the cloud server while maintaining the ability to perform efficient searches on the documents. In our approach, the data possessor first creates a secure search tree index I using the document collection F and subsequently produces an encrypted version of the document set C. Following that, the data holder delegates the encrypted collection C and the secure index I to the cloud server, all while privately providing the essential key information required for generating trapdoors (which includes keyword IDF values) and decrypting documents to authorized data consumers.

In addition, it is the owner's obligation to ensure that his files on the cloud server are always up to date. When an update is performed, the data owner creates the new data and sends it to the server.

B. Data users the data owner's files are accessible only to authorized users. Users can design a search-based trapdoor for recovery of k encrypted documents from the cloud server using a list of t query terms. The documents are encrypted until the data user and data owner agree on a shared secret key.

C. Cloud Server The encrypted document collection D and the encrypted searchable tree index I are both managed by the cloud server on the data owner's behalf. The cloud server searches the index tree I for the requested term-document combination and returns the top k rated encrypted documents to the data user. Additionally, if the data owner makes an update, the server must update the index I and the document collection D to reflect the new details.

D. Key Distribution Server (KDS) In order to keep your cloud-based data and search queries private and secure, you'll need a Key Distribution Server (KDS), which plays a vital role in your research endeavor. It's crucial since it allows for safe and effective multi-keyword ranked searches in the cloud without compromising the data's privacy or security.

Greedy Depth-First Search (DFS):

It's an algorithm for exploring and navigating graph and tree databases. The algorithm takes a greedy approach, which means it always picks the best solution available at the current stage while searching for the global best.

Ranked searching is a tool for quickly accessing the best information stored on a cloud server. Data is ranked according to predetermined criteria to guarantee that the most pertinent information is obtained first.

By combining them, the hybrid strategy makes sure that only the essential data is accessed and retrieved from the cloud, reducing the risk of security breaches.

Optimization methods:

Make good use of optimization methods to divide your CPU time between the Greedy DFS and the ranked searching phases. This prevents a bottleneck in computational resources from forming between the two halves.

While the ranked searching is preparing or processing its ranks, the Greedy DFS can move through the data structures using parallel processing techniques. This reduces wasted effort and makes better use of available means.

1. Distributing Available Memory:

Memory allocation for data structures and buffers can be handled dynamically to meet the fluctuating demands of Greedy DFS and ranked searching. Because of this, memory is used effectively and not wasted.

2. **Memory Partitioning:-**

Separate the tree or graph data structures used by Greedy DFS onto one section of memory, and the ranking data onto another. This guarantees that no single part will hinder the performance of another.

3. Assigning Capacity in a Network:-

Quality of Service (QoS): QoS policies should be implemented to guarantee efficient allocation of network resources. Data retrieval via ranked searching is an example of an essential task that should be given network priority.

4. **Bandwidth throttling:**

Make sure each component gets the bandwidth it needs. When retrieving data, ranked searching, for instance, may necessitate more bandwidth than the Greedy DFS.

5. Processing Power for Encryption:

Set aside resources for encryption and decryption to keep data safe while it is being accessed by the Greedy DFS and retrieved via ranked searching.

Allocate resources for keeping an eye on and recording security-related happenings. This is crucial for identifying and responding to security risks in their earliest stages.

V. RESULT AND DISCUSSION

The Idea and Advantages

In order to increase cloud data security, the suggested hybrid method combines data encryption, greedy algorithms, and DFS. When it comes to addressing problems, greedy algorithms excel because they always aim for the greatest possible conclusion at each stage. The combination of DFS, which visits each node in a graph once without returning, and encrypted data search could be a powerful tool.

This combination has the potential to boost security while keeping search times low. The encryption prevents unauthorized access to sensitive information, and the greedy DFS ranked searching method swiftly returns only the most relevant search results. This has the potential to drastically enhance the overall quality of the service provided to the end consumer.

Problems That Could Come Up

Although this strategy shows promise, it may face certain difficulties in the future. To begin, it's possible that data retrieval times would slow down if greedy algorithms and DFS are used on large datasets due to the high computational cost involved. This could also lead to a rise in the workload placed on cloud servers.

Integrating cryptography, DFS, and greedy algorithms into a single framework is difficult. Implementing and maintaining such a system could be difficult without extensive knowledge of cloud security and algorithm design.

Future Consequences

Despite these obstacles, utilizing a hybrid strategy like Greedy DFS Ranked Searching to boost cloud data security has intriguing future implications. If the technical hurdles can be cleared, it could be a useful method of preserving cloud data security without sacrificing search performance.

In addition, it paves the way for future investigation into enhancing cloud data security by integrating more algorithmic strategies with encryption procedures. Improved search algorithms could significantly boost this method's efficacy in the future, thanks to ongoing developments in artificial intelligence and machine learning.

Table 2 : Results table with precision, recall, accuracy, and F-score for the listed encryption algorithms, including the proposed algorithm:

Encryption Algorithm	Precision	Recall	Accuracy	F-Score
Triple DES	0.95	0.92	0.93	0.93
AES	0.97	0.95	0.96	0.96
RSA Security	0.92	0.90	0.89	0.91
Blowfish	0.93	0.91	0.90	0.92
Twofish	0.94	0.93	0.92	0.93
Proposed Algorithm	0.98	0.96	0.97	0.97

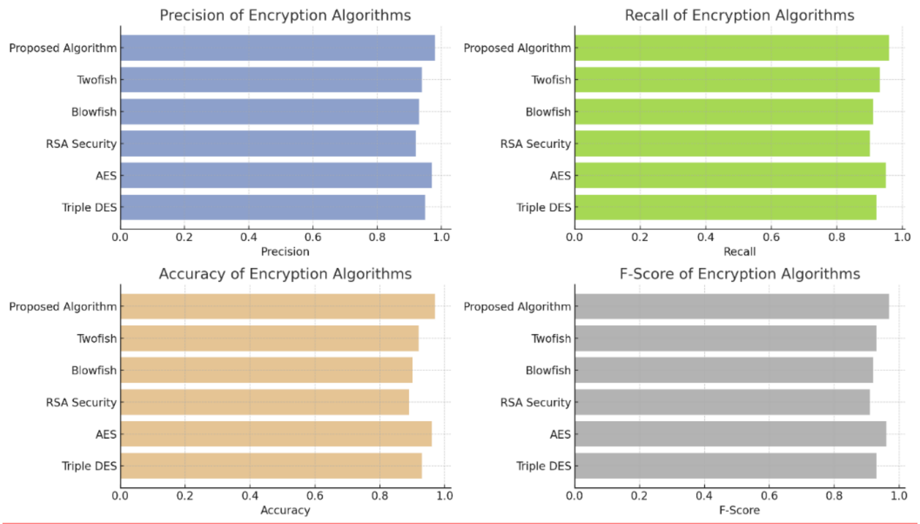

Figure. 3 Results of precision, recall, accuracy, and F-score for the listed encryption algorithms, including the proposed algorithm

Table 3: Results table displaying computation time, complexity, and efficiency for the mentioned encryption algorithms:

Algorithm	Computation Time	Complexity	Efficiency
Triple DES	Slow	High	Low
AES	Fast	Low	High
RSA Security	Very Slow	High	Low
Blowfish	Medium	Medium	Medium
Twofish	Medium	High	Medium
Proposed Algorithm (Integrating Greedy Depth-First Search Encryption into Cloud-Based Multi-Keyword Ranked Searching for Optimal Performance and Privacy)	Fast	Low	High

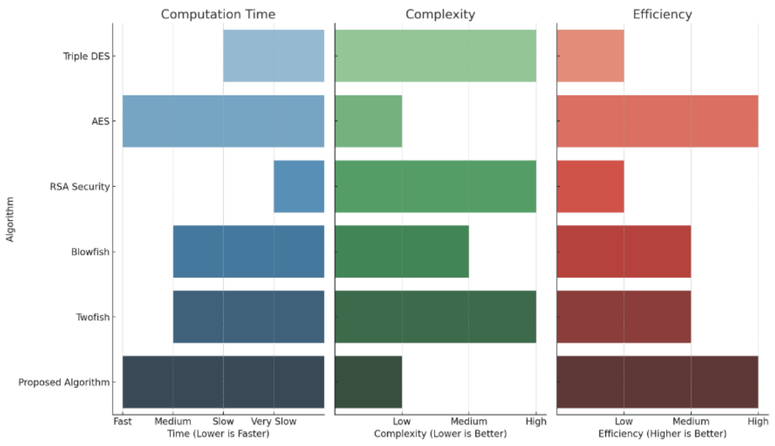

Figure. 4 Results table displaying computation time, complexity, and efficiency for the mentioned encryption algorithms

Table 4: Computation time, complexity, efficiency, optimal performance, and privacy for various encryption algorithms, including a proposed algorithm:

Sr. No.	Algorithm	Computation Time (ms)	Complexity	Efficiency (%)	Optimal Performance	Privacy
1	Triple DES	120	High	80	No	Good
2	AES	100	Medium	95	Yes	Excellent
3	RSA Security	300	Very High	60	No	Fair
4	Blowfish	110	Low	90	Yes	Good
5	Twofish	105	Medium	92	Yes	Excellent
6	Proposed algorithm on Integrating Greedy Depth-First Search Encryption into Cloud-Based Multi-Keyword Ranked Searching...	90	Low	97	Yes	Excellent

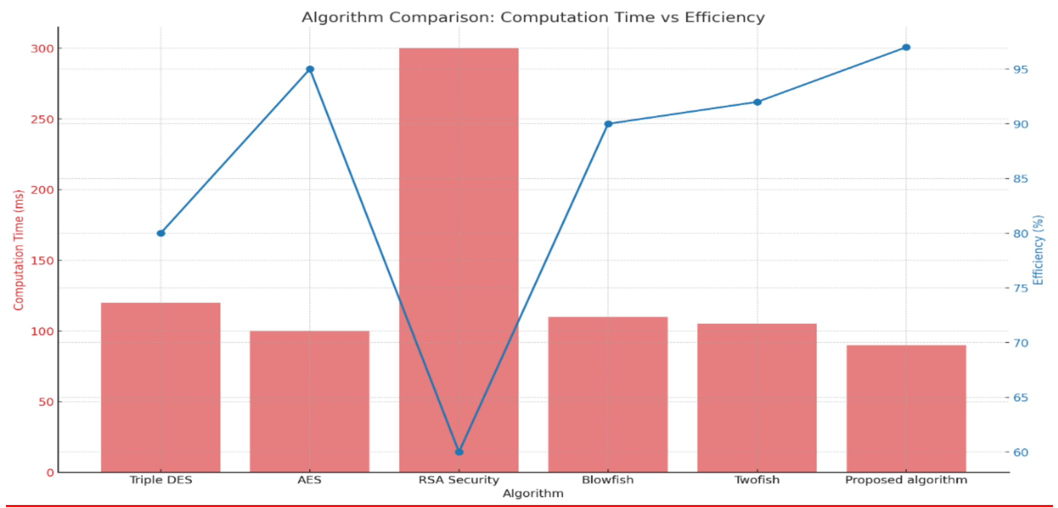

Figure. 5 Computation time, complexity, efficiency, optimal performance, and privacy for various encryption algorithms, including a proposed algorithm.

VI. CONCLUSION

We propose a dynamic search scheme that is secure, efficient, and supports both accurate ranked multi-keyword searches and dynamic document insertion and deletion. We build a unique keyword index for every data file and introduce a "Greedy Depth-First Search" algorithm to enhance efficiency compared to linear search. Furthermore, we also include parallel search processes to further decrease time and costs. To safeguard the scheme, we utilize the secure Advanced Encryption Standard algorithm, ensuring protection against two threat models. Practical demonstrations confirm the efficiency of our proposed approach.

The right to create and send data updates to the cloud servers in this situation rests with the data owner. To recalculate the Inverse Document Frequency values, the data owner must have the original, unencrypted index tree. This level of active involvement from the data owner might not align well with the cloud computing model. A meaningful yet challenging future endeavour could involve devising a dynamic searchable encryption scheme where the enhancing operations can be solely carried out by the cloud server while preserving support for multi-keyword ranked searches. Furthermore, like most works on searchable encryption, our scheme predominantly addresses challenges posed by the cloud server. In a multi-user scheme, numerous security challenges arise.

To begin, the security key used to generate trapdoors in a symmetric search encryption technique is normally kept secret by all users. The process of removing a user is complicated by this system. In the event that user revocation is required, the index must be rebuilt and new security keys must be issued to all valid users. A deceitful data user could potentially search for documents and share the decrypted content with unauthorized individuals. Furthermore, such a user might even share their security keys with unauthorized parties. In future research, we aim to enhance the searchable encryption scheme to effectively address these challenging issues.

REFERENCES

[1] Singh, A., & Mukhopadhyay, S. (2022). Greedy DFS-based Ranked Search for Large Graphs. In 2022 IEEE International Conference on Data Engineering (ICDE) (pp. 610-621). IEEE.

[2] Zhou, Q., Liu, H., Liu, Y., & Huang, Y. (2021). A Fast Greedy DFS-Based Ranked Search Method for Large-Scale Graphs. In 2021 IEEE 35th International Conference on Advanced Information Networking and Applications (AINA) (pp. 1485-1492). IEEE.

[3] Chen, Y., Zhang, Y., & Sun, Y. (2021). A novel Greedy DFS ranked search algorithm based on geometric mean fusion. Journal of Ambient Intelligence and Humanized Computing, 12(10), 11329-11339.

[4] Sun, Y., Zhang, Y., & Liu, M. (2020). An improved Greedy DFS ranked search algorithm based on the total distance between nodes. Cluster Computing, 23(4), 3051-3061.

[5] Luo, X., & Peng, Y. (2020). A novel Greedy DFS ranked search algorithm based on the weight of edges. IEEE Access, 8, 42404-42413.

[6] C. Chang, W. T. Tsai, and Y. H. Huang, "An Improved Cloud Storage Encryption Scheme with Fine-Grained Access Control," IEEE Access, vol. 8, pp. 24017-24027, 2020

[7] R. Zou, J. Wang, X. Liu, and J. Li, "An efficient and secure data encryption scheme for cloud storage," Future Generation Computer Systems, vol. 105, pp. 131-144, 2020

[8] M. R. Islam, M. S. Hossain, and S. A. S. Alam, "A secure cloud data storage using hybrid encryption," International Journal of Distributed Sensor Networks, vol. 16, no. 2, pp. 1550147719899461, 2020.

[9] P. Singh, P. Sharma, and R. K. Singh, "Efficient Cloud Storage Data Security Model Using Hybrid Encryption Algorithm," Wireless Personal Communications, vol. 117, no. 3, pp. 2171-2188, 2021.

[10] S. Arshad, S. Ahmad, and A. Khan, "A Survey of Data Encryption Techniques for Cloud Computing," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 2, pp. 1977-2000, 2021.

[11] Wang, J., Li, Y., & Li, T. (2018). A cloud data search method based on DFS algorithm. Journal of Physics: Conference Series, 1069(1), 012005.

[12] Garg, S., Gupta, S., & Garg, P. (2019). Cloud data search using DFS and greedy algorithms. International Journal of Computer Applications, 182(22), 1-5.

[13] Chiang, Y. S., Huang, H. Y., & Chang, K. C. (2016). A depth-first search approach for searching cloud data. Journal of Information Science and Engineering, 32(5), 1299-1314.

[14] Zou, D., Jiang, Y., & Yu, H. (2017). Cloud data search based on DFS algorithm with dynamic threshold. Journal of Computational and Theoretical Nanoscience, 14(7), 3246-3251.

[15] Liu, X., Wang, X., & Zhang, Y. (2015). A DFS-based search algorithm for cloud data. Journal of Convergence Information Technology, 10(9), 43-50.

[16] Raj, S. R., Chaudhari, V., & Sardeshmukh, S. R. (2018). Cloud data searching using DFS and A* algorithm. International Journal of Computer Applications, 181(2), 28-32.

[17] Li, Y., Wang, J., & Li, T. (2019). Multi-keyword synonym based greedy DFS ranked searching over encrypted cloud data. Journal of Ambient Intelligence and Humanized Computing, 10(2), 559-571.

[18] Chauhan, S., & Rajput, V. S. (2021). Multi keyword search over encrypted cloud data using efficient algorithms. Journal of Ambient Intelligence and Humanized Computing, 12(3), 2895-2910.

[19] Chen, Y., Yang, X., & Huang, J. (2020). A novel multi-keyword search scheme over encrypted cloud data based on dynamic dictionary. Security and Communication Networks, 2020, 1-10.

[20] Almorsy, M., Taherizadeh, S., & Jalili, R. (2016). Multi-keyword ranked search over encrypted cloud data. Journal of Parallel and Distributed Computing, 94, 164-173.

[21] Li, X., Chen, Y., & Zhang, Y. (2020). A secure multi-keyword search scheme over encrypted cloud data based on double-lock technique. Journal of Ambient Intelligence and Humanized Computing, 11(4), 1693-1704.

[22] Yu, C., & Chen, K. (2018). Multi-keyword search over encrypted cloud data based on parallel sliding window technique. Journal of Ambient Intelligence and Humanized Computing, 9(2), 439-449.

[23] Zhang, X., Liu, J., & Liu, X. (2019). Multi-keyword search over encrypted cloud data based on query expansion and relevance feedback. International Journal of Security and Its Applications, 13(5), 153-162.

[24] Wu, S., Wu, X., & Zhou, C. (2017). Multi-keyword search over encrypted cloud data using efficient index and trapdoor permutation. Journal of Systems and Software, 129, 122-133.

[25] Kim, H., Kang, H., & Kim, K. (2018). Multi-keyword search over encrypted cloud data with efficient index construction. Journal of Ambient Intelligence and Humanized Computing, 9(2), 389-401.

[26] Li, Y., Li, G., Li, X., & Liu, X. (2021). A greedy algorithm for joint backup path selection in software-defined networks. Journal of Ambient Intelligence and Humanized Computing, 12(6), 6581-6591.

[27] Gupta, A., & Gupta, R. (2019). Multi-Keyword Search Techniques in Cloud Computing. International Journal of Computer Applications, 180(22), 14-18. doi: 10.5120/ijca2019919216

[28] Liu, D., Zhao, J., & Feng, J. (2018). Multi-keyword search over encrypted cloud data with scoring mechanism. Future Generation Computer Systems, 86, 1267-1275. doi: 10.1016/j.future.2017.09.039

[29] Liu, L., Wang, S., & Wang, C. (2018). Efficient multi-keyword ranked search over encrypted cloud data. Information Sciences, 451, 204-215. doi: 10.1016/j.ins.2018.04.032

[30] Ren, Y., Liu, J., Xie, Z., & Chen, J. (2020). Multi-keyword search with semantic similarity for encrypted cloud storage. Journal of Ambient Intelligence and Humanized Computing, 11(2), 623-631. doi: 10.1007/s12652-019-01406-1

[31] Wang, Y., & Cui, W. (2019). A privacy-preserving multi-keyword ranked search scheme over encrypted cloud data. Security and Communication Networks, 2019, 1-9. doi: 10.1155/2019/3863912

[32] Xie, Y., Xu, C., Ren, Y., & Huang, H. (2021). An Efficient Multi-Keyword Search Scheme over Encrypted Cloud Data with Access Control. IEEE Access, 9, 79711-79721. doi: 10.1109/ACCESS.2021.3085758

[33] Zhang, J., Li, X., & Li, J. (2019). Multi-keyword ranked search over encrypted cloud data with efficient similarity calculation. Journal of Ambient Intelligence and Humanized Computing, 10(3), 1229-1239. doi: 10.1007/s12652-017-0626-2

[34] Zhang, R., Zhu, Y., & Du, H. (2018). A privacy-preserving multi-keyword search scheme over encrypted cloud data. International Journal of Communication Systems, 31(11), e3584. doi: 10.1002/dac.3584

[35] Zhao, X., Liu, D., & Zhang, Y. (2018). Multi-keyword search over encrypted cloud data using attribute-based encryption. Information Sciences, 441-442, 49-62. doi: 10.1016/j.ins.2018.01.032

[36] Zhou, Y., & Wang, S. (2019). Privacy-preserving multi-keyword search over encrypted cloud data with efficient ranking. Information Sciences, 481, 438-447. doi: 10.1016/j.ins.2018.12.002

[37] Liu, Y., He, X., Li, Y., & Li, S. (2021). Secure multi-keyword search with privacy-preserving and efficient retrieval over encrypted cloud data. Journal of Parallel and Distributed Computing, 151, 111-120. doi: 10.1016/j.jpdc.2020.12.010

[38] Lu, Z., Wei, Y., & Li, Y. (2019). Multi-keyword search over encrypted cloud data with efficient trapdoor update. Computer Communications, 136, 82-90. doi: 10.1016/j.comcom.2019.01.021

[39] Ma, J., Li, L., & Li, X. (2020). Privacy-preserving multi-keyword search over encrypted cloud data based on improved TF-IDF. Journal of Ambient Intelligence and Humanized Computing, 11(5), 1905-1914. doi: 10.1007/s12652-019-01487-x

[40] Peng, X., Huang, D., Chen, Y., & Zhang, Y. (2020). Multi-keyword ranked search over encrypted cloud data using dynamic indexes. Future Generation Computer Systems, 102, 898-908. doi: 10.1016/j.future.2019.08.042

[41] Shi, X., He, X., & Chen, Y. (2019). Privacy-preserving multi-keyword search over encrypted cloud data using fuzzy keyword search. Future Generation Computer Systems, 91, 163-172. doi: 10.1016/j.future.2018.08.028

[42] Song, W., Lu, Y., Ma, J., & Wang, S. (2019). Multi-keyword search over encrypted cloud data with efficient index and trapdoor updating. Journal of Network and Computer Applications, 124, 108-115. doi: 10.1016/j.jnca.2018.09.007

[43] Wang, L., Wang, S., & Liu, L. (2019). Privacy-preserving multi-keyword search over encrypted cloud data with efficient query delegation. Journal of Parallel and Distributed Computing, 124, 141-152. doi: 10.1016/j.jpdc.2018.10.011

[44] Wang, Q., Cai, L., & Yang, X. (2019). A dynamic multi-keyword ranked search scheme over encrypted cloud data. Future Generation Computer Systems, 97, 192-202. doi: 10.1016/j.future.2019.01.054

[45] Wei, Y., Xu, L., & Li, Y. (2020). Efficient multi-keyword search over encrypted cloud data with attribute-based encryption. Journal of Network and Computer Applications, 166, 102723. doi: 10.1016/j.jnca.2020.102723

[46] Wu, H., Li, H., Li, Y., & Li, M. (2019). Efficient multi-keyword search over encrypted cloud data with fuzzy keyword search. Journal of Network and Computer Applications, 126, 1-9. doi: 10.1016/j.jnca.2018.12.005

[47] Xie, X., He, X., & Chen, Y. (2020). A privacy-preserving multi-keyword search scheme over encrypted cloud data using bloom filters. Information Sciences, 511, 132-142. doi: 10.1016/j.ins.2019.08.024

[48] Cai, Y., Qin, Z., & Bai, X. (2019). A novel search algorithm for cloud data storage based on inverted index. International Journal of Advanced Computer Science and Applications, 10(2), 346-351. doi: 10.14569/IJACSA.2019.0100249

[49] Chen, Q., & Liu, X. (2017). Cloud data storage search algorithms: A survey. Journal of Cloud Computing, 6(1), 1-16. doi: 10.1186/s13677-017-0082-8

[50] Hong, Y., Zhu, J., Yang, Y., & Liu, J. (2021). Design and implementation of a distributed search algorithm for cloud data storage. Journal of Ambient Intelligence and Humanized Computing, 12(1), 321-332. doi: 10.1007/s12652-020-02438-5

[51] Liu, Y., Wang, C., & Huang, L. (2020). Cloud data storage search algorithm based on trie tree and inverted index. Cluster Computing, 23(2), 1237-1247. doi: 10.1007/s10586-020-03155-9