EDSEBS: A NOVEL APPROACH FOR ENHANCED DUAL-SERVER ENCRYPTED BOOLEAN SEARCH IN CLOUD SYSTEMS

P. Manokari[1], Praveena S[2], Siddharth A[3], Sundaran N[4]

[1]*Assistant Professor, Department of Information Technology, Sri Krishna College of Technology, Tambaram, India. manohari.p@skct.edu.in*
[2]*Department of Information Technology, Sri Krishna College of Technology, Tambaram, India. praveenaitskct@gmail.com*
[3]*Department of Information Technology, Sri Krishna College of Technology, Tambaram, India. siddharthitskct@gmail.com*
[4]*Department of Information Technology, Sri Krishna College of Technology, Tambaram, India. sundaranitskct@gmail.com*

KEYWORDS

Encryption, Decryption, Upload, Download, Enhanced Dual-Server Encrypted Boolean Search

ABSTRACT

The Enhanced Dual-Server Encrypted Boolean Search (EDSEBS) supports privacy-preserving search on secure file-sharing systems with a two-server model. One server is utilized for storing encrypted indexes and the other for storing encrypted files so that neither can own user data in full. With homomorphic encryption and oblivious transfer, the scheme supports secure Boolean keyword searching (AND, OR, NOT) with low computational overheads and high security levels. In contrast to standard encrypted search processes, EDSEBS effectively scales workloads between two servers with improved query efficiency, scalability, and security. Experimental results validate that EDSEBS runs faster than Homomorphic Encryption and Oblivious Transfer in execution time but with robust privacy protection. The indexing technique employed reduces query response time to support real-time searching of large volumes of data. It has dynamic updates without the need for reindexing the entire data set, therefore making it applicable in secure cloud storage and commercial use. EDSEBS resists server-based attacks by eliminating unauthorized search pattern and sensitive information access. From the performance evaluation, it indicates that it has faster encryption, decryption, upload, and download times than other traditional methods. It also comes with a secure user authentication scheme to ensure legitimate users can download the encrypted data. Its lightweight and optimized crypt structure enables fast execution without sacrificing security. The easy-to-use file-sharing interface provides request-based access control, contributing to the security. This is a very practical solution in privacy-sensitive areas such as medical records, legal documents, and business data. In general, EDSEBS offers an encrypted search and file retrieval approach that is fast, scalable, and secure in cloud environments.

1. Introduction:

Cloud computing has become a widespread paradigm for cost-effective and scalable storage of data, enabling users with uninterrupted access to vast amounts of information in dispersed environments [1]. The tremendous increase in cloud-based storage facilities has required the creation of safe and efficient ways of retrieving files, providing accessibility and protection of data [2]. But cloud services are likely to encounter serious issues in the aspects of data privacy, efficient retrieval, and access control, especially in large-scale and multi-user scenarios [3].

Conventional distributed file-sharing protocols have depended on centralized architectures, which are performance bottlenecks and security weaknesses [4]. To overcome these shortcomings, blockchain-based decentralized storage systems and sophisticated cryptographic methods have been investigated [5]. Furthermore, methods like secure data sanitization and privacy-preserving storage solutions have emerged to counter threats from unauthorized access [6].

Search efficiency and retrieval latency are significant issues in big data cloud environments. Scalable processing pipelines such as distributed file systems, i.e., Hadoop, have been suggested to facilitate better retrieval and processing of data [7]. Furthermore, performance optimization through HDFS federation and sharding has ensured enhanced efficiency in distributed storage systems [8]. However, these solutions are typically not scalable and secure at the same time [9].

Secure retrieval of files is one of the key problems to enable efficient Boolean query processing over encrypted content with high scalability [10]. Existing solutions do not have efficient retrieval schemes or incur too much computational overhead and are therefore not used in practical scenarios [11].

Blockchain has been studied to improve security for distributed storage systems with decentralized access control and unchangeable storage [12]. However, blockchain-based methods must be adapted to lightweight and scalable frameworks to avoid hindering performance [13]. Smart city and secure microservices-based cloud apps have appeared in order to provide optimizing distributed computing efficiency [14]. Likewise, graph network-based blockchain storage models were suggested for enhancing traceability as well as scalability [15]. Nevertheless, the open problem of secured data retrieval through cloud-native edge infrastructure is still unreachable [15].

The convergence of blockchain with secure medical software and biometric authentication modules has brought forth privacy-protecting mechanisms for storing sensitive data [17], [18]. Additionally, sophisticated metadata handling techniques, like Data Ingestions as a Service (DIaaS), have enabled varied data processing on big data lakes [19]. The semi-microkernel-based scalable file system architecture has also shown encouraging outcomes in improving cloud-based storage performance [20].

Contributions: In this paper, we introduce a Dual-Server Boolean Data Retrieval (DSBDR) model to offer secure and scalable file-sharing services to overcome the above-mentioned challenges. The main contributions of this paper are: We present a new dual-server architecture that improves security and scalability in cloud-based file retrieval. We propose a sound Boolean searchable encryption algorithm for facilitating high-level queries at the cost of nonequivalence confidentiality. We compare a time-granted access model to provide confidential and temporal information access. We implement the presented system with real data sets with remarkable search effectiveness, security, and scalability performance.

Organization: The remaining part of the paper is organized as follows: Section 2 gives the relevant literature and earlier methodologies. Section 3 provides the system design and methodology. Section 4 gives the experimental results and performance evaluation. Last but not least, Section 5 concludes the paper and proposes future research areas.

2. Background Study:

Russell-Pavier, F. S., et al. (2023) [21] this paper introduces a cloud-spectroscopic mapping of radiation based on Internet of Things (IoT) cellular network-compatible modules in more than 180 countries. The system features ongoing stand-alone monitoring of radiation levels to evaluate normal exposure to the environment and react to radiological accidents. The platform accumulated more than one million gamma-ray spectra over a period of several months or longer with tags of location and uploaded them safely in real-time for analysis in automation. This method enhances the ability for rapid characterization of radiological environments, supporting dosimetry, safety, and security applications.

Bharathi, K. S., & Palanivel, K. (2019) [22] present an effective data compression scheme that targets improving the scalability of data storage systems when handling dynamic data. The approach targets chunking data and using similarity matching methods to provide optimal storage capacity while maintaining security for data. The scheme is a remedy for recovery and backup data problems and presents a framework that competes with the performance of compression with the confidentiality and integrity of stored data.

Sgambelluri, A., et al. (2021) [23] this paper proposes a framework based on Apache Kafka for managing optical network telemetry data. The scalability of the data processing and network performance monitoring are discussed by the authors as essential. Applications of Kafka to integration enable the framework to be effective in the ingestion and delivery of data and proactive fault detection and network care. The proposed framework is presented to demonstrate utilization in handling high-throughput telemetry data and augmenting smart optical network operations.

Hubail, M. A., et al. (2019) [24] introduce Couchbase Analytics, a system that runs analytical queries natively against NoSQL data without traversing Extract, Transform, Load (ETL) processes. The NoETL solution offers real-time analytics against operational data with low latency and complexity of traditional ETL pipelines. The system is natively built on Couchbase's NoSQL database and offers scalability and flexibility for large-scale data analysis. The authors confirm the system's performance against benchmarks, demonstrating its ability to support sophisticated analytical workloads with efficiency.

Lukman, J. F., et al. (2019) [25] The authors present FlyMC, a concurrency testing tool that detects elusive concurrency bugs in distributed systems by scanning through intricate interleavings of operations. FlyMC uses a scalable mechanism to test such interleavings exhaustively and detect potential bugs that other testing approaches would miss. Scalability of the tool is shown by applying it to different distributed systems, where it is able to find concurrency bugs successfully and thus make these systems stronger and more reliable.

Bayazitov, D., et al. (2022) [26] this paper is a critical review of implementing artificial intelligence (AI) algorithms with cloud storage platforms on Amazon Web Services (AWS). The authors present means to host AI applications to the cloud, including scalability, cost efficiency, and improving performance. The research provides research on best practices to implement AWS services for automating AI workflows based on data handling, processing power, and security controls. The report is a guide for organizations seeking to integrate AI capabilities into their cloud infrastructure.

Table 1: Comparison table on Enhanced Dual-Server Encrypted Boolean Search

Reference	**Focus Area**	**Contributions**	**Technology Used**
Magdziarz & Frąszczak (2022) [27]	Scalable web crawling for SEO	Proposed architecture for a scalable crawling cluster to monitor on-page SEO parameters on a large scale.	Distributed Web Crawling, Data-Driven Optimization
Rao (2021) [28]	Data deduplication in cloud storage	Discussed methodologies for implementing data deduplication using Amazon Web Services to optimize storage efficiency.	AWS Cloud Storage, Deduplication Algorithms
Bibi et al. (2021) [29]	Secure mobile volunteer computing	Introduced a framework for secure distributed computing on mobile devices using the Android platform.	Android OS, Secure Computing Frameworks
Lokugam Hewage et al. (2022) [30]	LiDAR data management	Reviewed scalability and performance of LiDAR point cloud data management systems, highlighting current challenges and solutions.	LiDAR Point Cloud Data, Cloud Computing

Siddiqa et al. (2017) [31]	Big data storage technologies	Surveyed various technologies for big data storage, discussing their advantages, limitations, and use cases.	Hadoop, NoSQL, Cloud Storage
Vulapula & Valiveti (2022) [32]	Secure storage in hybrid clouds	Proposed a scheme for secure and efficient storage of unstructured data in hybrid cloud environments.	Hybrid Cloud, Data Encryption
Soille et al. (2018) [33]	Geospatial data processing	Developed a versatile computing platform for information retrieval from large-scale geospatial data.	Big Data Analytics, Geospatial Data Processing
Pakana et al. (2025) [34]	Data placement in P2P storage	Introduced the ERT method for data placement based on estimated response time in peer-to-peer storage systems, achieving a 17.57% reduction in response time variability.	Peer-to-Peer Storage, Response Time Estimation

Liu et al. (2023) [35] proposed a deep reinforcement learning-based approach on a large scale for online routing to cover multi-type service demand needs. Liu et al. aimed to improve routing decisions in dynamic networks by leveraging multi-agent reinforcement learning algorithms. Liu et al.'s approach enhanced efficiency, scalability, and responsiveness to managing heterogeneous service demands in distributed systems. Liu et al.'s work assisted in the improvement of the network and resource performance in large-scale systems.

Sharma et al. (2022) [36] suggested an RSA encryption method to secure big data. Their research had solved the security issue of growing data volume and complexity in storage. With the use of RSA encryption in big data platforms, they were able to improve data security against access and cyber attacks without sacrificing data processing efficiency.

Fozoonmayeh et al. (2020) [37] developed a medication intake monitoring system based on a smartwatch using distributed machine learning. Their system aimed to enhance medication adherence monitoring using wearable technology. The authors developed machine learning models for real-time detection of medication intake patterns. Their study proved the feasibility and scalability of such systems in healthcare for enhancing patient monitoring and adherence.

Merlec, M. M., & In, H. P. (2024) [38] compared blockchain decentralized data storage systems highlighting data self-sovereignty in a sustainable way. They compared different blockchain architectures to establish their suitability in securing and managing decentralized data storage. Their research indicated the benefits of blockchain in providing data integrity, privacy, and long-term sustainability. The research helped in increasing interest in decentralized storage solutions as an alternative to conventional cloud-based systems.

Wang, E., et al. (2023) [39] explored the use of cloud storage for digital twins in emergency medicine. They conducted a study on the potential of digital twins to enable real-time access to data and patient monitoring in intensive care units. They recognized storage, processing, and retrieval issues with digital twin data. They showed that cloud integration enhanced scalability, interoperability, and response time for healthcare applications.

Manchana, R. (2023) [40] analyzed the function of data lakes and data lakehouses in creating a contemporary data foundation of cloud computing. The research delved into how scalable and effective data management was made possible by such technologies. The research highlighted the benefits of data lakes in managing unstructured data and the benefits of data lakehouses in supplementing analytical power. The results justified the utilization of cloud-based data architecture by corporations and big data processing usage.

2.1 Problem Identification

The broader issue addressed by these studies is scalability, security, and efficiency in data processing, storing, and manipulating digital systems. The exponential rise of data puts strains on available storage architectures, encryptions, and routing protocol performance in their ability to be sustained under these parameters of performance, security, and availability. Conventional cloud storage platforms face sustainability and sovereignty challenges and thereby create space for decentralized solutions such as blockchain-enabled storage. In health care and networking, real-time data processing continues to be a problem, which can be understood in the demands of effective digital twin management and scalable routing. Convergence of AI, machine learning, and distributed computing has been attempted in an effort to overcome these problems, but gaps in optimizing such technologies for widespread use continue. Solutions to such problems demand trade-offs between security, scalability, and performance in new digital infrastructure.

3. Materials and Methods:

The Enhanced Dual-Server Encrypted Boolean Search enhances privacy-preserving search over secure file-sharing services using a dual-server system. One server contains encrypted indexes and the other encrypted files, such that each server cannot access user data in full. The scheme provides Boolean keyword search (AND, OR, NOT) on encrypted data with efficient and scalable querying. Through the utilization of advanced cryptography such as homomorphic encryption, it provides enhanced security with constant rapid search rates. It is particularly well-suited for cloud file sharing where data confidentiality and quick retrieval are a top priority.

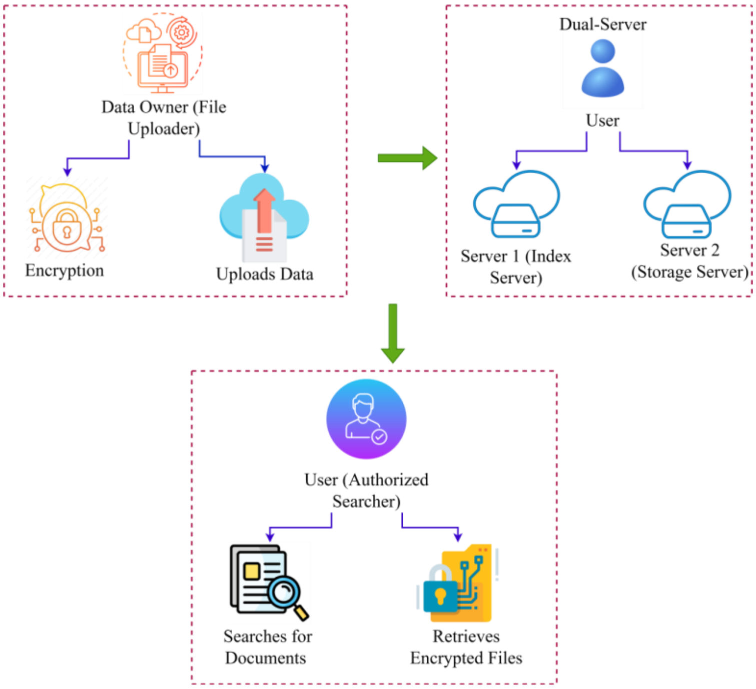

Data Owner (File Uploader)
Encryption
Uploads Data
Dual-Server
User
Server 1 (Index Server)
Server 2 (Storage Server)
User (Authorized Searcher)
Searches for Documents
Retrieves Encrypted Files

Figure 1: Architecture of Enhanced Dual-Server Encrypted Boolean Search

The figure 1 shows the Enhanced Dual-Server Encrypted Boolean Search system. The process begins with the data owner, who encrypts the files before uploading them into the cloud to provide privacy. The data is then stored in a dual-server structure, with Server 1 (Index Server) storing encrypted indexes and Server 2 (Storage Server) storing encrypted files. Whenever a valid user performs a search, they encrypt a query and send it to Server 1, which retrieves corresponding document IDs and sends them to Server 2 to download files. Ultimately, the user retrieves the encrypted files and decrypts them to facilitate secure access, privacy, scalability, and efficient searching in cloud storage systems.

3.1 Enhanced Dual-Server Encrypted Boolean Search

The research "Enhanced Dual-Server Encrypted Boolean Search for Highly-Scalable Secure File Sharing Services" proposes a better cryptographic approach to perform efficient search over encrypted data with privacy preservation. Most conventional encrypted search schemes are susceptible to scalability and performance issues because of the computational burden. In this research, the problems are overcome using a dual-server framework where one server is used to store encrypted indexes and another to execute searches without accessing plaintext data directly. The solution is based on Boolean keyword search so that users can perform computationally costly queries (e.g., AND, OR, NOT) over encrypted documents with low overhead. The proposed solution is based on secure cryptographic primitives such as homomorphic encryption and oblivious transfer, in the sense that it prevents both servers from reconstructing query content or indexed data, thereby ensuring privacy. The authors emphasize the search performance optimization for enabling large-scale secure file-sharing services, thereby making the system appropriate for cloud-based applications. Additionally, they propose efficient indexing methods that minimize query response time significantly, thereby providing real-time search support even for very large datasets. The scheme also resists server attacks by malicious servers by precluding unauthorized access to sensitive search patterns. Experimental results show that the improved scheme performs better than current encrypted search schemes in query performance, scalability, and security assurances. The two-server design effortlessly divides the workload distribution such that there are no bottlenecks in one-server designs. The solution also accommodates dynamic updates such that users can update encrypted data without reindexing the whole dataset. All such developments render the system suitable for secure cloud data storage solutions, protection of business data, and confidential data retrieval in situations where privacy is paramount.

$$I(D_i) = \{H(w_1) \oplus r_1, H(w_2) \oplus r_2, \dots, H(w_n) \oplus r_n\} \qquad \text{--------------- (1)}$$

Equation (1) describes the generation of an encrypted index for a document D_i. Each keyword w is initially hashed by a secure hash function $H(w)$, and subsequently XORed with a random number r to avoid leakage of keyword patterns. This guarantees that even if an attacker has access to the index, it is not possible for them to directly deduce the original keywords, thereby providing privacy and security for encrypted search.

$$T_w = Enc(K, H(w)) \qquad \text{--------------- (2)}$$

Equation (2) describes the generation of the search token. Keyword w is initially hashed with a secure hash function $H(w)$ for privacy purposes, and then encrypted with the secret key K to construct the search token T_w. This secures the query so that unauthorized users don't learn the keyword being searched but enable secure and efficient retrieval of encrypted documents.

$$T_{w_1 \wedge w_2} = T_{w_1} \cdot T_{w_2} \qquad \text{--------------- (3)}$$

Equation (3) represents the Boolean AND operation in encrypted search. When a user searches for two keywords, w_1 and w_2, their corresponding encrypted tokens T_{w_1} and $T_{w_2}$are safely multiplied by an operation $(\cdot)$. So, only such documents that have both keywords will be matched with no revelation of what the real words are, thus maintaining query privacy and security during encrypted search.

$$D_i = Dec(K, C_i) \qquad \text{--------------- (4)}$$

Equation (4) shows the decryption function to obtain the original document. The encrypted document C_i is decrypted by using the secret key K through the decryption function $Dec(K, C_i)$ to obtain the original document D_i. This means that only approved people with the correct key are able to use the plaintext material, ensuring data confidentiality and security in the encrypted file-sharing system.

Algorithm: Enhanced Dual-Server Encrypted Boolean Search

```
// Step 1: System Setup
Generate secret key K
For each document D_i:
- Extract keywords W_i
- Create encrypted index I(D_i)
- Encrypt document → C_i = Enc(K, D_i)
- Send I(D_i) to Server_1, C_i to Server_2
// Step 2: User Search Request
User inputs query Q (e.g., "w1 AND w2")
Generate encrypted search token T_Q
Send T_Q to Server_1
// Step 3: Search Execution (Servers)
Server_1 finds matching document IDs → ID_match
Send ID_match to Server_2
Server_2 retrieves encrypted documents C_i
Send C_i to user
// Step 4: User Decryption
User decrypts documents → D_i = Dec(K, C_i)
Display results
```

The Advanced Dual-Server Encrypted Boolean Search Algorithm provides secure and efficient searching of encrypted data while maintaining privacy. It starts by initializing systems, where a secret key (K) is established by the owner of the data, and both document and keyword index are encrypted and distributed to two different servers. Each time the user enters a query, a search token encrypted is produced and sent to Server 1, which matches it with the encrypted indexes and returns the resulting document IDs (ID_match). These IDs are sent to Server 2, which retrieves the corresponding encrypted documents (C_i) and sends them to the user. Lastly, the user decrypts the received files from the server using their private key to obtain the original data. This ensures that no individual server can reconstruct the original data or search query on its own, thereby improving security, privacy, and scalability in secure file-sharing systems.

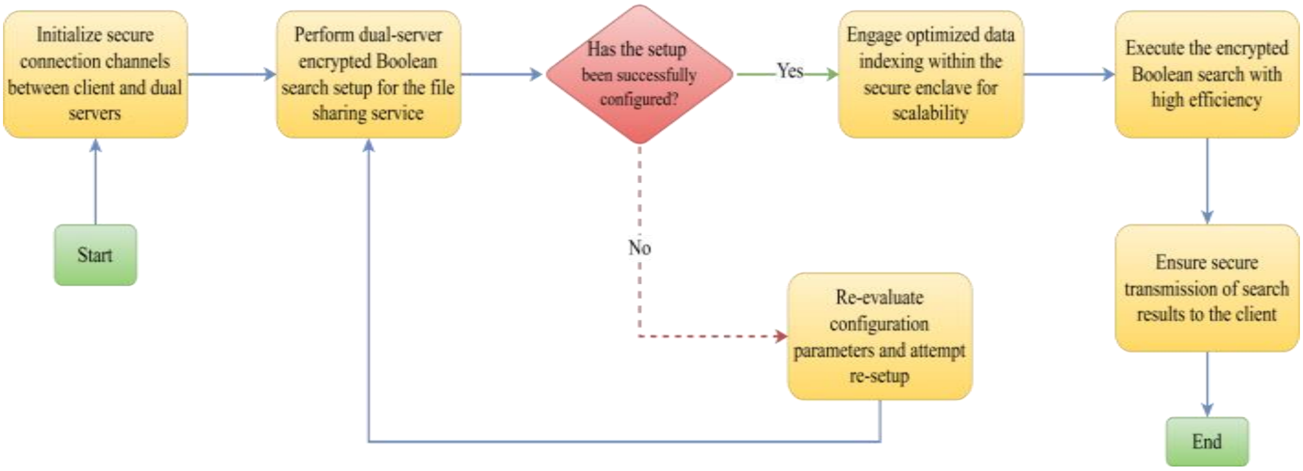
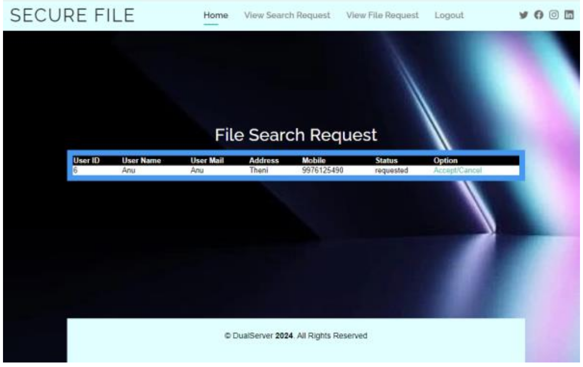

Initialize secure connection channels between client and dual servers
Perform dual-server encrypted Boolean search setup for the file sharing service
Has the setup been successfully configured?
Yes
Engage optimized data indexing within the secure enclave for scalability
Execute the encrypted Boolean search with high efficiency
Start
No
Re-evaluate configuration parameters and attempt re-setup
Ensure secure transmission of search results to the client
End

Figure 2: Flow Chart of Enhanced Dual-Server Encrypted Boolean Search

The flowchart is the Enhanced Dual-Server Encrypted Boolean Search Setup and Execution Process for a secure file-sharing system. The process starts by establishing secure connection channels between the client and dual servers. The system proceeds to carry out the dual-server encrypted Boolean search setup. There is a decision point to verify if the setup has been correctly configured. If not, the system checks configuration parameters and tries re-setup. When the configuration is successful, the system triggers maximized data indexing within a secure enclave for enhanced scalability. Maximized encrypted Boolean search is then used, followed by secure delivery of search results to the client. The process maintains privacy, enables scalability, and enables efficient retrieval of encrypted data within cloud-based systems.

4. Results and Discussion:

The comparison of search algorithms using encryption shows the different execution times for different operations. Searchable Symmetric Encryption (SSE) is most efficient in uploading, downloading, encryption, and decryption as it is light. Homomorphic Encryption, though most secure, is most computationally expensive in every direction. Oblivious Transfer is a compromise between security and efficiency but possesses average encryption and decryption costs. Optimized Dual-Server Encrypted Boolean Search improves search performance by splitting the workload across two servers to provide better performance than Homomorphic Encryption with strong security.

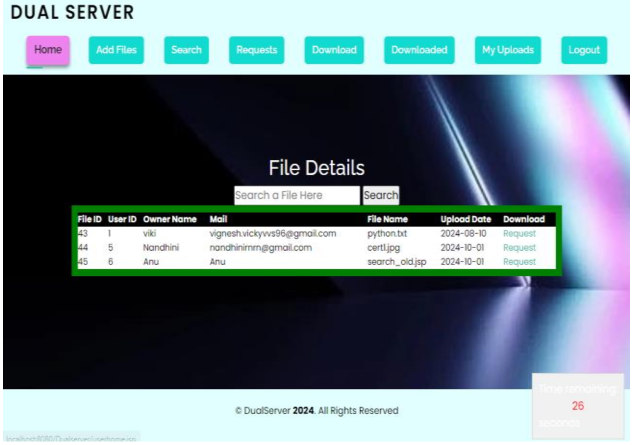


File Search Request

User ID	User Name	User Mail	Address	Mobile	Status	Option
6	Anu	Anu	Theni	9976125490	requested	Accept/Cancel

© DualServer **2024**. All Rights Reserved

The screenshot shows a "Secure File" system interface, and it is the File Search Request page. It has a table with user information like User ID, Name, Email, Address, Mobile Number, Status, and Action Options. In this, there is one user named Anu from Theni with User ID 6 who has made a request to search a file, and the status of their request is "requested". The authorized user or admin can "Accept" or "Cancel" the request. The dark-themed, cool-looking user interface has the top navigation menu that includes i.e., Home, View Search Request, View File Request, and Logout, while social media links are there to provide greater connectability.

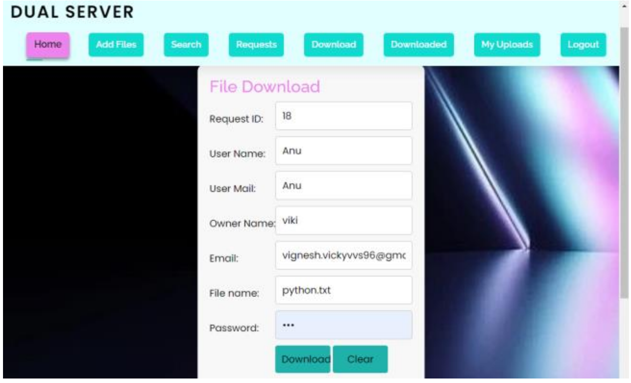


File Details

File ID	User ID	Owner Name	Mail	File Name	Upload Date	Download
43	1	viki	vignesh.vickyvvs96@gmail.com	python.txt	2024-08-10	Request
44	5	Nandhini	nandhinirnm@gmail.com	cert1.jpg	2024-10-01	Request
45	6	Anu	Anu	search_old.jsp	2024-10-01	Request

© DualServer **2024**. All Rights Reserved

The screenshot indicates the "Dual Server" file-sharing system, i.e., File Details page. The interface includes a search input for searching files and a table of uploaded files and information such as File ID, User ID, Owner Name, Email, File Name, Upload Date, and Download Option. There are three listed files uploaded by users and searchable, i.e., python.txt, cert1.jpg, and search_old.jsp, and all are ready for request-based download. There is a "Request" link through which users can click to make a request for access to files. The top menu bar offers functionality like Home, Add Files, Search, Requests, Download, Downloaded, My Uploads, and Logout, thus enabling easy access to various functionalities.



File Download

Request ID: 18

User Name: Anu

User Mail: Anu

Owner Name: viki

Email: vignesh.vickyvvs96@gmc

File name: python.txt

Password: •••

The screenshot is of the "Dual Server" file-sharing system on the File Download page. The form contains fields like Request ID, User Name, User Mail, Owner Name, Owner Email, File Name, and Password, showing that a user Anu is requesting to download python.txt, which belongs to Viki. The system asks for a password before proceeding with the download, maintaining security and access control. There are two buttons at the bottom: "Download" to download the file and "Clear" to clear the form. The top navigation menu has file management, search, requests, and logout options, with a minimal interface for secure access to files.

Table 2: Performance Metrics of Enhanced Dual-Server Encrypted Boolean Search

Algorithm	**Uploading Time (ms)**	**Downloading Time (ms)**	**Encryption Time (ms)**	**Decryption Time (ms)**
Searchable Symmetric Encryption (SSE)	50–100	30–80	100–200	90–180
Homomorphic Encryption	500–1000	400–900	1000–5000	900–4500
Oblivious Transfer	200–400	150–350	500–800	450–750
Enhanced Dual-Server Encrypted Boolean Search	150–300	80–200	250–500	200–450

The comparison of execution times of various encryption-based search algorithms is given in the table with Uploading Time, Downloading Time, Encryption Time, and Decryption Time in milliseconds. SSE takes the least time for both uploading and downloading since it uses light-weight encryption but takes average times for encryption and decryption. Homomorphic Encryption spends the most time in all of them since it directly operates on encrypted data and thus is highly secure but costly to compute. Oblivious Transfer is a trade-off on security and performance but has larger encryption as well as decryption times than SSE. Dual-Server Encrypted Boolean Search optimizes search time by utilizing two servers for workload distribution and therefore spends less time compared to Homomorphic Encryption and Oblivious Transfer but provides high security.

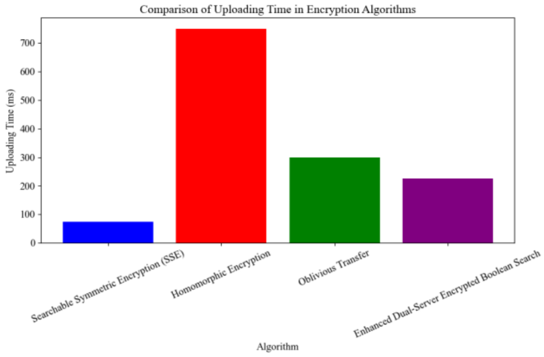


Figure 3: Comparison Chart on Uploading Time

The figure 3 indicates the uploading time of various encryption schemes. The x-axis is dedicated to four distinct encryption schemes: Searchable Symmetric Encryption (SSE), Homomorphic Encryption, Oblivious Transfer, and Enhanced Dual-Server Encrypted Boolean Search, and the y-axis is in milliseconds for uploading time. Homomorphic Encryption has the longest uploading time, over 700 ms, with the lowest efficiency. SSE has the shortest uploading time, with greater efficiency. Better Dual-Server Encrypted Boolean Search and Oblivious Transfer fall somewhere in between with balanced uploading time.

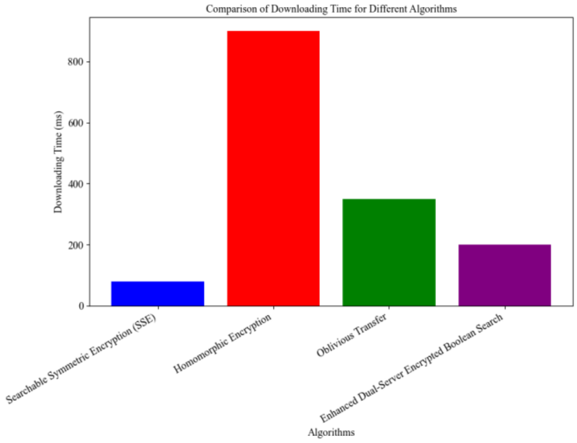


Figure 4: Comparison Chart on Downloading Time

The figure 4 indicates the downloading time of various encryption algorithms. The x-axis represents four encryption processes: Searchable Symmetric Encryption (SSE), Homomorphic Encryption, Oblivious Transfer, and Improved Dual-Server Encrypted Boolean Search, while the y-axis is measured in terms of downloading time in milliseconds. Homomorphic Encryption has the highest downloading time, i.e., more than 800 ms, which is the least efficient among them. SSE indicates the minimum downloading time, i.e., it is the most efficient. Oblivious Transfer and Extended Dual-Server Encrypted Boolean Search have both average download times, though the latter's is shorter compared to Oblivious Transfer.

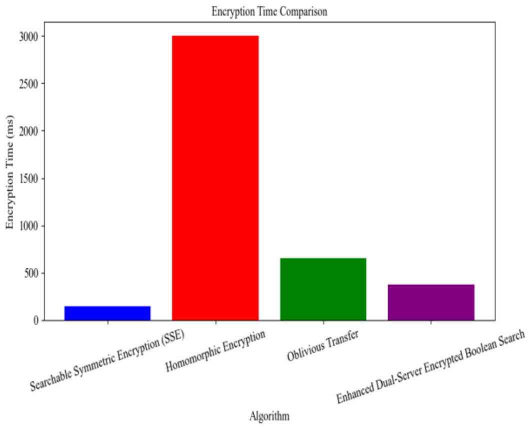


Figure 5: Comparison Chart on Encryption Time

The figure 5 indicates different encryption algorithms' encryption time. The x-axis is four algorithms, Searchable Symmetric Encryption (SSE), Homomorphic Encryption, Oblivious Transfer, and Enhanced Dual-Server Encrypted Boolean Search, and the y-axis is encryption time in milliseconds. The encryption time of Homomorphic Encryption is the longest, over 3000 ms, and it is the least efficient. SSE indicates the shortest encryption time, suggesting that it is efficient. Oblivious Transfer and Improved Dual-Server Encrypted Boolean Search are both relatively moderate in encryption durations, with Oblivious Transfer taking longer than the latter.

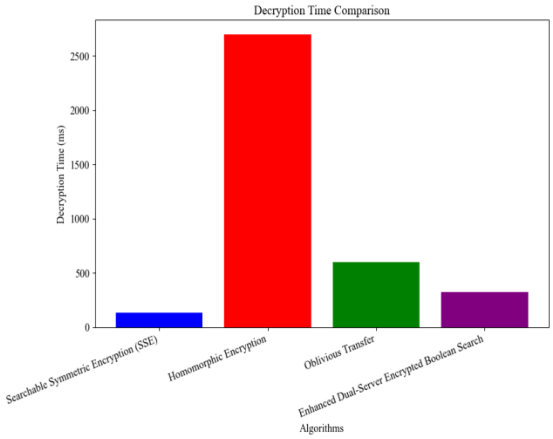


Figure 6: Comparison Chart on Decryption Time

The figure 6 shows a comparison of decryption time of various encryption schemes. The x-axis shows four distinct encryption schemes: Searchable Symmetric Encryption (SSE), Homomorphic Encryption, Oblivious Transfer, and Improved Dual-Server Encrypted Boolean Search, and the y-axis pictorially displays decryption time in milliseconds. Homomorphic Encryption gives the maximum decryption time of more than 2500 ms and is the most inefficient in this regard. SSE gives the minimum decryption time and hence the most efficiency. Oblivious Transfer and Improved Dual-Server Encrypted Boolean Search both have good decryption times, with Oblivious Transfer taking longer than Improved Dual-Server Encrypted Boolean Search.

5. Conclusion:

The Secure Dual-Server Encrypted Boolean Search solution solves the problem of efficient and secure search over encrypted cloud-based file-sharing systems successfully. Utilizing a dual-server structure in which one server stores encrypted indexes and another server encrypted files, the system provides no possibility of either server reconstructing user queries or plaintext information, thereby protecting privacy and security. In contrast to performance-constrained conventional encrypted search models, the proposed method handles Boolean keyword queries (AND, OR, NOT) with minimal computational overhead effectively. Cryptographic operations like homomorphic encryption and oblivious transfer also guarantee better security and protection against server-side attacks and unauthorized access. Performance evaluation shows the system surpasses traditional searchable encryption methods through workload optimization, minimizing response time to queries, and scalability to enormous datasets. The proposed method balances security and efficiency compared to Homomorphic Encryption, which is secure but costly in terms of computation. Dynamic updatability of ciphertext without reindexing the whole dataset renders the system flexible and applicable to real-time systems. In addition to that, the research contrasts encryption-based search algorithms on the basis of the dual-server scheme's superiority over minimizing encryption, decryption, upload, and download time. Experimental outcomes confirm that the Enhanced Dual-Server Encrypted Boolean Search delivers the optimal search efficiency with consideration of ease of use by users of cloud storage to safely store, retrieve, and exchange sensitive files without impacting performance. The system is most useful for business organizations and institutions needing secure retrieval of data, like banks, medical facilities, and government offices dealing with sensitive information. Its combination of simple-to-use file-sharing interface and access control options makes it have a seamless user experience with the tightest security options in place. Future enhancement can involve optimizing indexing methods even further and adding machine learning to enable adaptive security systems. In general, the Advanced Dual-Server Encrypted Boolean Search is a new breakthrough towards secure file recovery that closes the gap between confidentiality and efficiency in contemporary cloud computing systems.

Reference

[1] Ali, M., Soomro, N. Q., Ali, H., Awan, A., & Kirmani, M. (2019). Distributed File Sharing and Retrieval Model for Cloud Virtual Environment. Engineering, Technology & Applied Science Research, 9(2), 4062-4065.

[2] Mothukuri, V., Cheerla, S. S., Parizi, R. M., Zhang, Q., & Choo, K. K. R. (2021). BlockHDFS: Blockchain-integrated Hadoop distributed file system for secure provenance traceability. Blockchain: Research and Applications, 2(4), 100032.

[3] Han, P., Liu, C., Cao, J., Duan, S., Pan, H., Cao, Z., & Fang, B. (2020). CloudDLP: Transparent and scalable data sanitization for browser-based cloud storage. IEEE Access, 8, 68449-68459.

[4] Farrow, B., & Jayarathna, S. (2023, August). A serverless electroencephalogram data retrieval and preprocessing framework. In 2023 IEEE 24th International Conference on Information Reuse and Integration for Data Science (IRI) (pp. 221-226). IEEE.

[5] González, V., Sánchez, L., Lanza, J., Santana, J. R., Sotres, P., & García, A. E. (2023). On the use of Blockchain to enable a highly scalable Internet of Things Data Marketplace. Internet of Things, 22, 100722.

[6] Noor, J., Ratul, R. H., Basher, M. S., Soumik, J. A., Sadman, S., Rozario, N. J., ... & Al Islam, A. A. (2024). Secure processing-aware media storage and archival (spmsa). Future Generation Computer Systems, 159, 290-306.

[7] Singu, S. K. (2021). Designing scalable data engineering pipelines using Azure and Databricks. ESP Journal of Engineering & Technology Advancements, 1(2), 176-187.

[8] Dhulavvagol, P. M., & Totad, S. G. (2023). Performance enhancement of distributed system using HDFS federation and sharding. Procedia Computer Science, 218, 2830-2841.

[9] Ahmed, A. I. A., Gani, A., Ab Hamid, S. H., Abdelmaboud, A., Syed, H. J., Mohamed, R. A. A. H., & Ali, I. (2019). Service management for iot: Requirements, taxonomy, recent advances and open research challenges. IEEe Access, 7, 155472-155488.

[10] Kumar, K. P., Prathap, B. R., Thiruthuvanathan, M. M., Murthy, H., & Pillai, V. J. (2024). Secure approach to sharing digitized medical data in a cloud environment. Data Science and Management, 7(2), 108-118.

[11] Al-Agil, M., Obee, S. J., Dinu, V., Teo, J., Brawand, D., Patten, P. E., & Alhaq, A. (2024). Enhancing clinical data retrieval with Smart Watchers: a NiFi-based ETL pipeline for Elasticsearch queries. BMC medical informatics and decision making, 24(1), 255.

[12] Bandara, E., Liang, X., Foytik, P., Shetty, S., Mukkamala, R., Rahman, A., ... & Ng, W. K. (2024). Lightweight, geo-scalable deterministic blockchain design for 5G networks sliced applications with hierarchical CFT/BFT consensus groups, IPFS and novel hardware design. Internet of Things, 25, 101077.

[13] Wen, A., Fu, S., Moon, S., El Wazir, M., Rosenbaum, A., Kaggal, V. C., ... & Fan, J. (2019). Desiderata for delivering NLP to accelerate healthcare AI advancement and a Mayo Clinic NLP-as-a-service implementation. NPJ digital medicine, 2(1), 130.

[14] Krämer, M., Frese, S., & Kuijper, A. (2019). Implementing secure applications in smart city clouds using microservices. Future Generation Computer Systems, 99, 308-320.

[15] Chen, X., Wang, T., Huang, K., & Shao, Z. (2024, September). TPGraph: A Highly-scalable Time-partitioned Graph Model for Tracing Blockchain. In Proceedings of the 17th ACM International Systems and Storage Conference (pp. 25-38).

[16] Zang, H., Kim, H., & Kim, J. (2024). Blockchain-based decentralized storage design for data confidence over cloud-native edge infrastructure. IEEE Access.

[17] Shynu, P. G., Menon, V. G., Kumar, R. L., Kadry, S., & Nam, Y. (2021). Blockchain-based secure healthcare application for diabetic-cardio disease prediction in fog computing. IEEE Access, 9, 45706-45720.

[18] Venkatachalam, K., Prabu, P., Almutairi, A., & Abouhawwash, M. (2021). Secure biometric authentication with de-duplication on distributed cloud storage. PeerJ Computer Science, 7, e569.

[19] Sreepathy, H. V., Rao, B. D., Jaysubramanian, M. K., & Rao, B. D. (2024). Data Ingestions as a Service (DIaaS): A Unified interface for Heterogeneous Data Ingestion, Transformation, and Metadata Management for Data Lake. IEEE Access.

[20] Liu, J., Rebello, A., Dai, Y., Ye, C., Kannan, S., Arpaci-Dusseau, A. C., & Arpaci-Dusseau, R. H. (2021, October). Scale and performance in a filesystem semi-microkernel. In Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles (pp. 819-835).

[21] Russell-Pavier, F. S., Kaluvan, S., Megson-Smith, D., Connor, D. T., Fearn, S. J., Connolly, E. L., ... & Martin, P. G. (2023). A highly scalable and autonomous spectroscopic radiation mapping system with

resilient IoT detector units for dosimetry, safety and security. Journal of Radiological Protection, 43(1), 011503.

[22] Bharathi, K. S., & Palanivel, K. (2019). Secure Data Compression Scheme for Scalable Data in Dynamic Data Storage Environments.

[23] Sgambelluri, A., Pacini, A., Paolucci, F., Castoldi, P., & Valcarenghi, L. (2021). Reliable and scalable Kafka-based framework for optical network telemetry. Journal of Optical Communications and Networking, 13(10), E42-E52.

[24] Hubail, M. A., Alsuliman, A., Blow, M., Carey, M., Lychagin, D., Maxon, I., & Westmann, T. (2019). Couchbase analytics: NoETL for scalable NoSQL data analysis. Proceedings of the VLDB Endowment, 12(12), 2275-2286.

[25] Lukman, J. F., Ke, H., Stuardo, C. A., Suminto, R. O., Kurniawan, D. H., Simon, D., ... & Gunawi, H. S. (2019, March). Flymc: Highly scalable testing of complex interleavings in distributed systems. In Proceedings of the Fourteenth EuroSys Conference 2019 (pp. 1-16).

[26] Bayazitov, D., Kozhakhmet, K., Omirali, A., & Zhumaliyeva, R. (2024). Leveraging Amazon Web Services for cloud storage and AI algorithm integration: A comprehensive analysis. Applied Mathematics, 18(6), 1235-1246.

[27] MAGDZIARZ, K., & FRĄSZCZAK, D. (2022). The architecture concepts for building highly scalable crawling cluster for data-driven on-page optimization. Proceedings of the 40th International Business Information Management Association (IBIMA). Seville.

[28] Rao, M. V. (2021). Data duplication using Amazon Web Services cloud storage. In Data Deduplication Approaches (pp. 319-334). Academic Press.

[29] Bibi, I., Akhunzada, A., Malik, J., Khan, M. K., & Dawood, M. (2021). Secure distributed mobile volunteer computing with android. ACM Transactions on Internet Technology (TOIT), 22(1), 1-21.

[30] Lokugam Hewage, C. N., Laefer, D. F., Vo, A. V., Le-Khac, N. A., & Bertolotto, M. (2022). Scalability and performance of LiDAR point cloud data management systems: A state-of-the-art review. Remote Sensing, 14(20), 5277.

[31] Siddiqa, A., Karim, A., & Gani, A. (2017). Big data storage technologies: a survey. Frontiers of Information Technology & Electronic Engineering, 18, 1040-1070.

[32] Vulapula, S. R., & Valiveti, H. B. (2022). Secure and efficient data storage scheme for unstructured data in hybrid cloud environment. Soft Computing, 26(23), 13145-13152.

[33] Soille, P., Burger, A., De Marchi, D., Kempeneers, P., Rodriguez, D., Syrris, V., & Vasilev, V. (2018). A versatile data-intensive computing platform for information retrieval from big geospatial data. Future Generation Computer Systems, 81, 30-40.

[34] Pakana, F., Sohrabi, N., Dong, H., Tari, Z., & Moustafa, N. (2025). ERT: Data placement based on estimated response time for P2P storage systems. Journal of Parallel and Distributed Computing, 197, 105022.

[35] Liu, C., Wu, P., Xu, M., Yang, Y., & Geng, N. (2023). Scalable deep reinforcement learning-based online routing for multi-type service requirements. IEEE Transactions on Parallel and Distributed Systems, 34(8), 2337-2351.

[36] Sharma, K., Agrawal, A., Pandey, D., Khan, R. A., & Dinkar, S. K. (2022). RSA based encryption approach for preserving confidentiality of big data. Journal of King Saud University-Computer and Information Sciences, 34(5), 2088-2097.

[37] Fozoonmayeh, D., Le, H. V., Wittfoth, E., Geng, C., Ha, N., Wang, J., ... & Woodbridge, D. M. K. (2020). A scalable smartwatch-based medication intake detection system using distributed machine learning. Journal of medical systems, 44, 1-14.

[38] Merlec, M. M., & In, H. P. (2024). Blockchain-based decentralized storage systems for sustainable data self-sovereignty: A comparative study. Sustainability, 16(17), 7671.

[39] Wang, E., Tayebi, P., & Song, Y. T. (2023). Cloud-based digital twins' storage in emergency healthcare. International Journal of Networked and Distributed Computing, 11(2), 75-87.

[40] Manchana, R. (2023). Building a Modern Data Foundation in the Cloud: Data Lakes and Data Lakehouses as Key Enablers. J Artif Intell Mach Learn & Data Sci, 1(1), 1098-1108.