International Journal of Advanced Computing and Mechanical Systems (IJACM)
E-ISSN No.: xxxx-xxxx www.ijacm.com
Volume Number: 1 **Issue Number:** 1 **pp:** 1- 7 **Publication Month:** Dec-2025

Design of a Trapdoor-Enabled Symmetric Scheme for Secure Ciphertext Discovery

Ramya P[1], Rama Devi S[2,*]
[1]*Department of MCA, S A Engineering College, Thiruverkadu, Chennai, India*
[2]*Department of CSE, S A Engineering College, Thiruverkadu, Chennai, India*
Corresponding Author E-Mail: *ramadevisena*@gmail.com

ABSTRACT

This paper presents an efficient and lightweight symmetric searchable encryption (SSE) approach that enables string-based queries with only a single round of communication and low computational overhead on encrypted documents. Unlike traditional SSE methods that rely on multiple encryption operations for index construction, the proposed technique employs hash-chain–based indexes, making it suitable for environments with limited computational resources. Compared with existing SSE string search schemes, the information disclosed to the server is restricted to what can be inferred from previous searches, namely the frequency and relative positions of queried strings. A distinguishing feature of this work is the introduction of trapdoor mechanisms specifically designed for string searches within the SSE setting, which has not been addressed in prior studies. The primary objective is to provide formal proof of non-adaptive security against an honest but curious server under standard security definitions. In addition, the paper proposes a new security notion, referred to as search pattern privacy, which measures and limits information leakage caused by trapdoor exposure. The scheme's security is established under the notion of search pattern indistinguishability.

Keywords: Symmetric Searchable Encryption, String-Based Secure Search, Trapdoor Mechanisms, Search Pattern Privacy

I. INTRODUCTION

Symmetric Searchable Encryption (SSE) allows secure search operations over encrypted document collections while preventing the server from learning sensitive information about both the stored data and user queries. This is typically achieved by employing symmetric cryptographic primitives instead of computationally expensive public-key techniques, accepting a limited and controlled level of information leakage to improve efficiency.

In traditional searchable encryption schemes, string queries are commonly handled as multi-keyword searches, where a phrase is decomposed into individual words and processed independently. For example, a string may be treated as a set of keywords without preserving their adjacency or order. This limitation reduces the expressiveness of the search. The proposed string search approach overcomes this issue by explicitly considering the ordering of words, enabling more accurate and meaningful query results. A consistent example is used throughout the paper and is revisited in Section IV to explain the construction and execution of the proposed algorithms and data structures.

The proposed scheme achieves non-adaptive security by utilizing a sequence of hashing operations rather than chains of encryption, resulting in improved performance and suitability for lightweight

doi: https://doi.org/10.5281/zenodo.18212533

applications. Unlike keyword-centric approaches, the scheme supports efficient searching for arbitrary strings that may not be easily extracted as keywords, while intentionally allowing minimal leakage to maintain efficiency. Related work has explored phrase search and proximity-based queries over encrypted cloud data [1], as well as faster secure string search techniques using Bloom filters.

Our scheme completes search operations in a single communication round and requires optimal-time computation for searching a string across n documents. It imposes no storage overhead on the client and requires only linear storage on the server. Information leakage is minimized such that the server learns nothing directly about the frequency or relative positions of searched words beyond what can be inferred from the history of queries. Unlike earlier index construction methods based on repeated encryption operations, the proposed approach employs hash-chain techniques, making it more efficient for resource-constrained environments.

For the first time, this work addresses string search in SSE under an active adversarial model, where an attacker may insert malicious documents into the dataset. A modified version of the scheme is introduced to securely handle such adversaries, requiring two rounds of communication and limited client-side keyword storage. The practicality of the proposed solution is demonstrated through experiments conducted on two commercial datasets.

II. RELATED WORKS

Searchable encryption has been widely studied, with early research focusing on formal definitions and security properties. Abdalla et al. [1] revisited searchable encryption and examined its consistency properties, also exploring its relationship with anonymous identity-based encryption and functional extensions. Privacy-preserving search over encrypted cloud data has been addressed by several works. Cao et al. [2] proposed techniques for ranked multi-keyword search, enabling secure and efficient retrieval while preserving user privacy. Cash et al. [3] analysed leakage-abuse attacks in searchable encryption systems and identified potential vulnerabilities arising from access and search pattern leakage. In later work, Cash et al. [4] proposed data structures and implementation strategies to support dynamic searchable encryption for very large databases. Highly scalable SSE constructions supporting Boolean queries were introduced in [5], improving both expressiveness and performance. The locality properties of searchable symmetric encryption were further studied by Cash and Tessaro [6], focusing on efficiency implications related to data organization. Improved security definitions and efficient SSE constructions were proposed by Curtmola et al. [7], strengthening privacy guarantees. Dynamic SSE schemes supporting secure updates to encrypted data were explored by Kamara and Papamanthou [8]. Foundational cryptographic principles relevant to searchable encryption are comprehensively presented in standard cryptography references such as [9]. More recent efforts have focused on lightweight phrase search in cloud environments, as demonstrated by Li et al. [10], which emphasizes efficiency but differs from index-based constructions. In contrast to probabilistic or filter-based approaches, the present work adopts an index-based design and provides a formal proof of non-adaptive security under established definitions.

III. RESULTS AND DISCUSSION

This section presents the overall design, working principles, and functional components of the proposed symmetric searchable encryption–based string search system. The primary goal of the proposed framework is to enable efficient and privacy-preserving string search over encrypted data stored on an untrusted cloud server while minimizing computation, communication overhead, and information

doi: https://doi.org/10.5281/zenodo.18212533

leakage. A key contribution of this work is the introduction of search pattern security, which ensures that repeated queries do not reveal exploitable patterns to the server beyond what is inherently leaked through query history. The proposed scheme is proven secure under the search pattern indistinguishability definition. Unlike dynamic index structures, the index generated in this scheme is constructed once by the client during the initial setup phase and remains unchanged for the same dataset. This static nature significantly reduces system complexity and improves performance, especially in lightweight and resource-constrained environments.

In the proposed symmetric searchable encryption (SSE) model, the client encrypts the data locally before outsourcing it to the cloud server. The client is free to organize the encrypted documents in any structure and may maintain auxiliary metadata to improve retrieval efficiency. Although the initial preprocessing and index construction cost at the client side is proportional to the dataset size, subsequent search operations incur minimal computation for both the client and the cloud server.

3.1 System Architecture

The overall system architecture of the proposed scheme is illustrated in Fig. 1. The system consists of three main entities: the Data Owner, the Data User, and the Cloud Server. Each entity has a well-defined role to ensure confidentiality, integrity, and efficient retrieval of encrypted data.

- The **Data Owner** is responsible for data encryption, index generation, and access control.
- The **Data User** performs authorized searches over encrypted data using trapdoor keys.
- The **Cloud Server** stores encrypted documents and indexes and performs search operations without learning sensitive information.

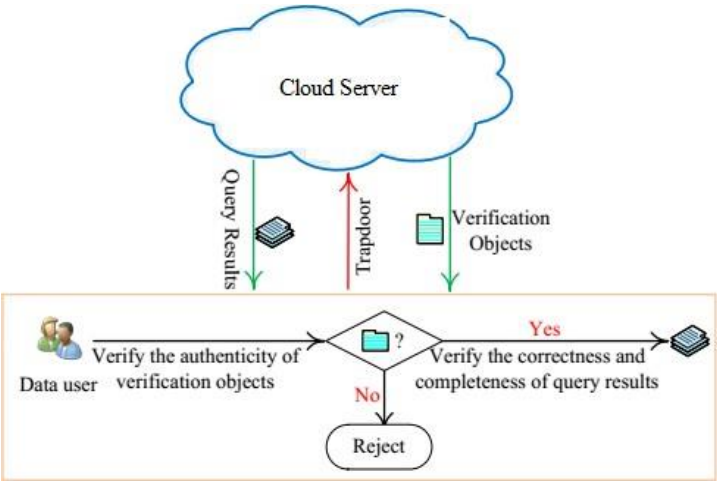
Cloud Server
Query Results
Trapdoor
Verification Objects
Data user
Verify the authenticity of verification objects
?
Yes
No
Verify the correctness and completeness of query results
Reject

Figure 1: System architecture of the proposed SSE-based string search scheme

As shown in Fig. 1, when a data user submits a search query, a trapdoor is generated and sent to the cloud server. The server executes the search operation over the encrypted index and returns both the query results and corresponding verification objects. The data user then verifies the authenticity, correctness, and completeness of the results before accepting them.

3.2 Data Owner Module

doi: https://doi.org/10.5281/zenodo.18212533

The Data Owner module manages data creation, encryption, upload, and access authorization. Initially, the data owner must register by providing valid credentials. After successful registration, the owner logs into the system through the secure login interface, as shown in Fig. 2.

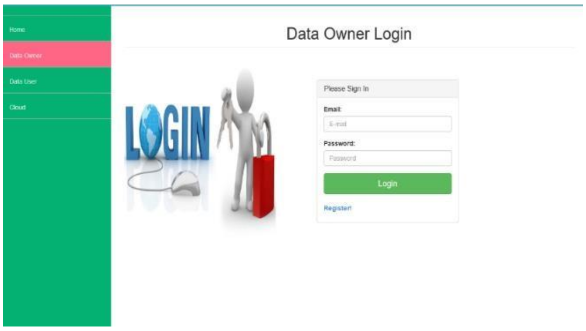

Figure 2: Data owner login page.

Once authenticated, the data owner can upload files to the cloud server. During the upload process, each file is encrypted locally using symmetric encryption, and associated keywords are processed using hash-based indexing techniques. This ensures that neither the plaintext content nor the keywords are exposed to the cloud server. The file upload interface is shown in Fig. 3. The data owner can view all uploaded files and monitor access requests sent by data users. When a data user requests access to a file, the data owner evaluates the request and either approves or rejects it. Upon approval, the data owner securely sends the trapdoor key, verification object, and decryption key to the authorized data user through a secure communication channel (e.g., registered email). This mechanism ensures fine-grained access control and prevents unauthorized data disclosure.

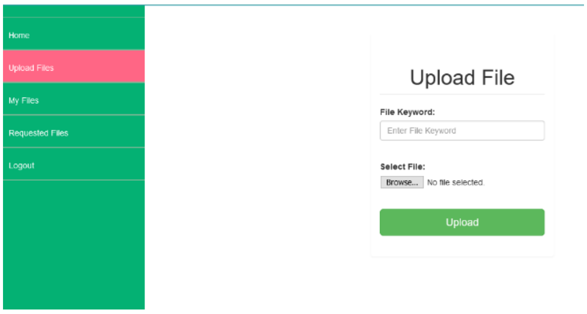

Figure 3: File upload interface for data owners

3.3 Data User Module

The Data User module allows authorized users to search and access encrypted files stored in the cloud. Like the data owner, a data user must first register and then log in using valid credentials. The login interface for data users is shown in Fig. 4. After successful authentication, the data user can search for files using string-based queries. The search interface, illustrated in Fig. 5, allows users to enter a keyword or string, which is transformed into a trapdoor using symmetric cryptographic primitives. This trapdoor is sent to the cloud server, ensuring that the actual query remains hidden. The data user can view the list of matching files returned by the cloud server and send access requests to the corresponding data owners. Once the request is approved, the user receives the trapdoor, verification object, and decryption key. Using these elements, the user verifies the correctness and completeness of the search results and decrypts the authorized files locally.

doi: https://doi.org/10.5281/zenodo.18212533

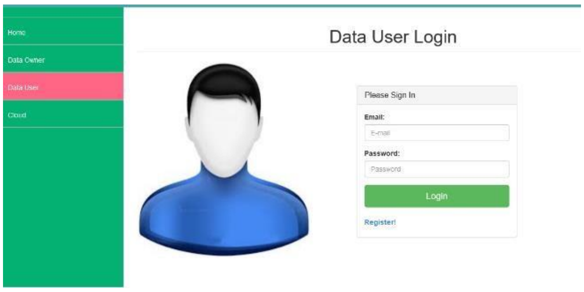

Figure 4: Data user login page

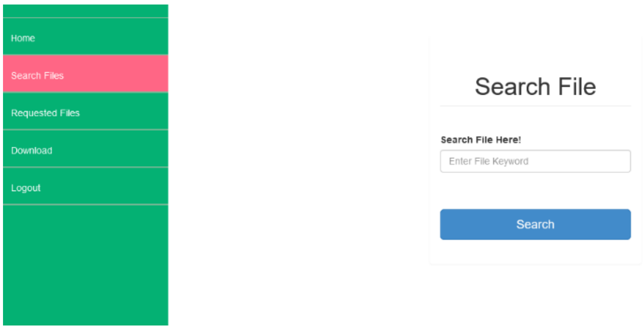

Figure 5: Secure file search interface for data users

3.4 Cloud Server Module

The Cloud Server (CS) acts as a storage and computation provider. It stores encrypted documents, encrypted indexes, and verification objects provided by data owners. The cloud server is considered honest-but-curious, meaning it follows the protocol correctly but may attempt to infer sensitive information from stored data and observed queries. The cloud server performs search operations using trapdoors submitted by data users. It returns the encrypted search results along with verification objects without learning the plaintext data, query content, or keyword relationships. The cloud server can also manage metadata related to data owners and users but does not possess decryption capabilities.

IV. RESULTS AND DISCUSSION

This section discusses the implementation results and evaluates the performance, security, and usability of the proposed system. The system was implemented using a web-based architecture, and experiments were conducted on two commercial datasets to validate practicality and efficiency.

4.1 Data Owner Login and Home Page

The Data Owner Login Page, shown in Fig. 2, provides a secure authentication interface for data owners. After successful login, the owner is redirected to the home page, where various functionalities such as file upload, file management, and access request handling are available. The interface is designed to be user-friendly while ensuring secure session management.

4.2 File Upload Functionality

The file upload interface, illustrated in Fig. 4, allows data owners to upload files along with encrypted keywords. During this process, the system performs encryption and hash-chain–based index generation locally. Experimental results show that the upload time increases linearly with file size, which is expected

doi: https://doi.org/10.5281/zenodo.18212533

due to encryption and indexing overhead. However, this overhead is incurred only once during data outsourcing.

4.3 Data User Login and Search Interface

The Data User Login Page, shown in Fig. 3, enables secure user authentication. After login, users can access the search interface displayed in Fig. 5. The search operation requires only a single round of communication between the user and the cloud server, significantly reducing latency compared to multi-round protocols. The use of hash-chain–based indexing ensures that search operations are computationally efficient. Experimental evaluation confirms that search time grows linearly with the number of documents, which is optimal for SSE-based string search systems.

4.4 Security and Verification Analysis

The proposed scheme ensures that the cloud server learns minimal information, limited to what can be inferred from past search history. The inclusion of verification objects enables data users to verify the correctness and completeness of query results, thereby preventing malicious behavior by the cloud server. Furthermore, the system supports resistance against active adversaries by allowing controlled verification and access mechanisms. Although the enhanced security variant requires additional communication rounds and limited client-side storage, it significantly improves robustness against document injection attacks.

The experimental results demonstrate that the proposed scheme achieves a favorable balance between efficiency and security. By avoiding repeated encryption operations and relying on hash-based indexing, the system is well suited for lightweight applications such as cloud storage services and resource-constrained environments. The results confirm that the proposed approach effectively supports secure string search with minimal leakage and practical performance.

V. CONCLUSION

This paper presented an efficient and lightweight symmetric searchable encryption scheme that enables secure string-based searches over encrypted data stored in an untrusted cloud environment. By leveraging hash-chain–based index construction instead of repeated encryption operations, the proposed approach significantly reduces computational overhead and is well suited for resource-constrained applications. The scheme supports single-round search communication and requires minimal client-side storage while maintaining optimal search efficiency across large document collections. A key contribution of this work is the introduction of search pattern privacy, which limits information leakage arising from repeated queries. The proposed system is formally shown to be secure under the notion of search pattern indistinguishability against an honest-but-curious server. Furthermore, the scheme addresses string search functionality by preserving the order of keywords, overcoming the limitations of traditional multi-keyword searchable encryption techniques. To enhance security, the work also considers an active adversarial model and proposes a modified construction capable of resisting document injection attacks, at the cost of additional communication rounds and limited client-side storage. Experimental evaluation on real-world datasets demonstrates the practicality, scalability, and effectiveness of the proposed scheme. Overall, the results confirm that the proposed solution provides a balanced trade-off between efficiency, functionality, and security for secure cloud-based string search applications.

doi: https://doi.org/10.5281/zenodo.18212533

VI. REFERENCES

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Tadayoshi Kohno, Tanja Lange and Haixia Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. volume 21, pages 350–391. Springer, 2008.

[2] Ning Cao, Cong Wang, Ming Li, Kui Ren, and Wenjing Lou. Privacy- Preserving Multi-Keyword Ranked Search Over Encrypted Cloud Data. volume 25, pages 222–233. IEEE, 2014.

[3] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage- Abuse Attacks Against Searchable Encryption. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pages 668–679. ACM, 2015.

[4] David Cash, Stanislaw Jarecki, Charanjit S Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation. volume 2014, page 853. Citeseer, 2014.

[5] David Cash, Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel- C˘at˘alin Ros¸u, and Michael Steiner. Highly-Scalable Searchable Symmetric Encryption With Support for Boolean Queries. In Advances in Cryptology–CRYPTO 2013, pages 353–373. Springer, 2013.

[6] David Cash and Stefano Tessaro. The Locality of Searchable Symmetric Encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 351–368. Springer, 2014.

[7] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. volume 19, pages 895–934. IOS Press, 2011.

[8] SenyKamara,Charalampos Papamanthou. Dynamic Searchable Symmetric Encryption. In Proceedings of the 2012 ACM conference on Computer and communications security, pages 965–976. ACM, 2012.

[9] Jonathan Katz and Yehuda Lindell. Introduction to Modern Cryptography. CRC press, 2014.

[10] Mingchu Li, Wei Jia, Cheng Guo, Weifeng Sun, and Xin: Lightweight Phrase Search With Symmetric Searchable Encryption in Cloud Storage. In Information Technology-New Generations (ITNG), 2015 12th International Conference on, pages 174–178. IEEE, 2015.

doi: https://doi.org/10.5281/zenodo.18212533