Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 16, Issue. 1: 2025
ISSN : **1906-9685**

Journal of Nonlinear Analysis and Optimization : Theory & Applications

Editors-in-Chief :

A SECURE SEARCHABLE ENCRYPTION MODEL FOR PRIVACY - CENTRIC CLOUD STORAGE

Mr. K. JAYA KRISHNA, P. PRABHU KUMAR

#1 Associate Professor Department of Master of Computer Applications
#2 Pursuing M.C.A
QIS COLLEGE OF ENGINEERING & TECHNOLOGY
Vengamukkapalem(V), Ongole, Prakasam dist., Andhra Pradesh- 523272

Abstract

Searchable encryption has received a significant attention from the research community with various constructions being proposed, each achieving asymptotically optimal complexity for specific metrics (e.g., search, update). Despite their elegance, the recent attacks and deployment efforts have shown that the optimal asymptotic complexity might not always imply practical performance, especially if the application demands a high privacy. In this article, we introduce a novel Dynamic Searchable Symmetric Encryption (DSSE) framework called Incidence Matrix (IM)-DSSE, which achieves a high level of privacy, efficient search/update, and low client storage with actual deployments on real cloud settings. We harness an incidence matrix along with two hash tables to create an encrypted index, on which both search and update operations can be performed effectively with minimal information leakage. This simple set of data structures surprisingly offers a high level of DSSE security while achieving practical performance. Specifically, IM-DSSE achieves forward privacy, backward-privacy and size-obliviousness simultaneously. We also create several DSSE variants, each offering different trade-offs that are suitable for different cloud applications and infrastructures. We fully implemented our framework and evaluated its performance on a real cloud system (Amazon EC2). We have released IM-DSSE as an open-source library for wide development and adaptation.

Introduction:

The rise of cloud storage and computing services provides vast benefits to the society and IT industry. One of the most important cloud services is data Storage-as-a-Service (SaaS), which can significantly reduce the cost of data management via continuous service, expertise and maintenance for resource-limited clients such as individuals or small/medium businesses. Despite its benefits, SaaS also brings significant security and privacy concerns to the user. That is, once a client outsource his/her own data to the cloud, sensitive information (e.g., email) might be exploited by a malicious party (e.g., malware). Although standard encryption schemes such as Advanced Encryption Standard (AES) can provide confidentiality, they also prevent the client from querying encrypted data from the cloud. This privacy versus data utilization dilemma may significantly degrade the benefits and usability of cloud systems. Therefore, it is vital to develop privacy-enhancing technologies that can address this problem while retaining the practicality of the underlying cloud service. Searchable Symmetric Encryption (SSE) [1] enables a client to encrypt data in such a way that they can later perform keyword searches on it. These encrypted queries are performed via "search tokens" [2] over an encrypted index which represents the relationship between search token (keywords) and encrypted files. A prominent application of SSE is to enable privacy-preserving keyword search on the cloud (e.g., Amazon S3), where a data owner can outsource a collection of encrypted files and perform keyword searches on it without revealing the file and query contents [3]. Preliminary SSE schemes (e.g., [1], [4]) only provide searchonly functionality on static data (i.e., no dynamism), which strictly limits their applicability due to the lack of update capacity. Later, several Dynamic Searchable Symmetric Encryption (DSSE) schemes (e.g., [3], [5]) were proposed thatpermit the user to add and delete files after the system is set up. To the best of our knowledge, there is no single DSSE scheme that outperforms all the other alternatives in terms of all the aforementioned metrics: privacy (e.g., information leakage), performance (e.g., search, update delay), storage efficiency and functionality. In the following, we first provide an overview on DSSE research and then, outline our research objectives and contributions toward addressing some of the limitations of the state-of-the-arts. SSE was first introduced by Song et al. [4]. Curtmola et al. [1] proposed a sublinear SSE scheme and introduced the security notion for SSE called adaptive security against chosen-keyword attacks (CKA2). Refinements of [1] have been proposed which offer extended functionalities (e.g., [6], [7]). However, the static nature of those schemes limited their applicability to applications that require dynamic file collections. Kamara et al. were among the first to develop a DSSE scheme in [3] that could handle dynamic file collections via an encrypted index. However, it leaks significant information for updates and it is not parallelizable. Although a number of DSSE schemes have been introduced the literature, most of them only provide a theoretical asymptotic analysis1 and, in some cases, merely a prototyp implementation. The lack of experimental performanc evaluations on real platforms poses a significant difficulty I assessing the application and practicality of proposed DSS schemes, as the impacts of security vulnerability, hidden computation costs, multi-round communication delay and storage blowup might be overlooked. For instance, most efficient DSSE schemes (e.g., [5], [10]) are vulnerable to file-injection attacks, which have been shown to be easily conducted even by a semi-honest adversary in practice, especially in the personal email scenario. Although several forward-secure DSSE schemes with an optimal asymptotic complexity have been proposed, they incur either very high delay due to public-key operations (e.g., [11]), or significant storage blow-up at both client and server side (e.g., [2]), and therefore, their ability to meet actual need of real systems in practice is still unclear.

Literature Survey:

A High-Security Searchable Encryption Framework for Privacy-Critical Cloud Storage Services

Authors: Thang Hoang, Attila A. Yavuz, Jorge Guajardo

Merits: Introduces the IM-DSSE framework, achieving forward and backward privacy with minimal information leakage. Demonstrates practical deployment on Amazon EC2, highlighting efficiency and scalability. Open-source implementation available for community use. Demerits: Limited support for complex queries beyond keyword-based searches. Potential performance overhead due to the use of incidence matrices and hash tables.

Hiding the Access Pattern is Not Enough: Exploiting Search Pattern Leakage in Searchable Encryption

Authors: Simon Oya, Florian Kerschbaum

Merits: Identifies the vulnerability of search pattern leakage in existing SSE schemes. Proposes an attack model that outperforms previous methods in keyword recovery. Demerits: Focuses primarily on theoretical analysis without proposing a concrete solution. Does not address the implementation challenges of mitigating search pattern leakage.

Obfuscated Access and Search Patterns in Searchable Encryption

Authors: Zhiwei Shang, Simon Oya, Andreas Peter, Florian Kerschbaum Merits: Introduces the OSSE scheme, obfuscating both access and search patterns to enhance privacy. Achieves efficient single-round communication with minimal client-side storage. Demerits: Limited scalability for large-scale datasets. Potential trade-off between privacy and system performance.

Dynamic Searchable Symmetric Encryption Schemes Supporting Range Queries with Forward/Backward Privacy

Authors: Cong Zuo, Shi-Feng Sun, Joseph K. Liu, Jun Shao, Josef Pieprzyk Merits: Proposes DSSE schemes supporting range queries with forward and backward privacy. Addresses file-injection attacks and content leakage of deleted documents.

Demerits: Supports only a limited number of documents in certain schemes. Security proofs are provided in the random oracle model, which may not reflect real-world conditions

Functional Requirements

1. Data Encryption Before Upload o All files must be encrypted on the client side before being uploaded to the cloud.

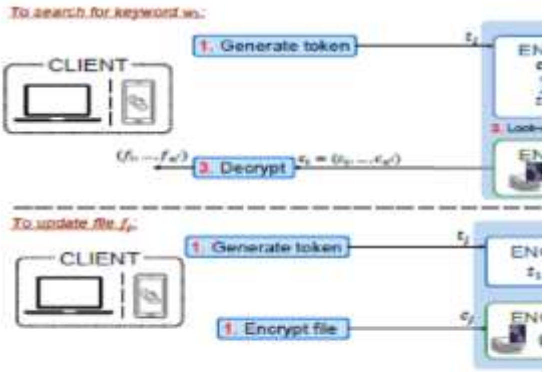
2. Searchable Encryption Support o Users can perform keyword or multi-keyword searches over encrypted data without decrypting it.

3. Secure Index Creation o Create encrypted indexes (e.g., inverted index, bloom filters) for efficient search capabilities.

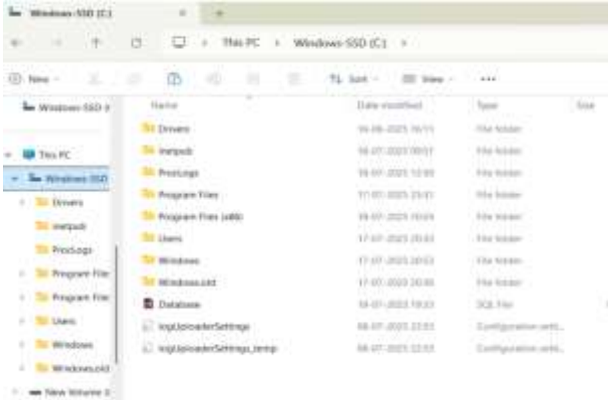
4. Query Generation & Encryption o Allow users to encrypt their search terms/queries locally before sending them to the cloud.

5. Query Execution Over Encrypted Index o The cloud server must match encrypted queries against the encrypted index and return matching encrypted results. 6. Result Decryption on Client-Side Users should be able to decrypt search results locally using their private keys.

System Architecture:

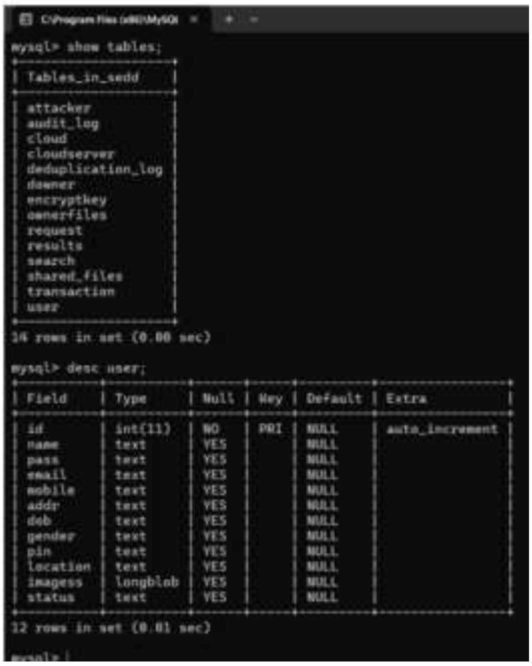


SCREENSHOTS



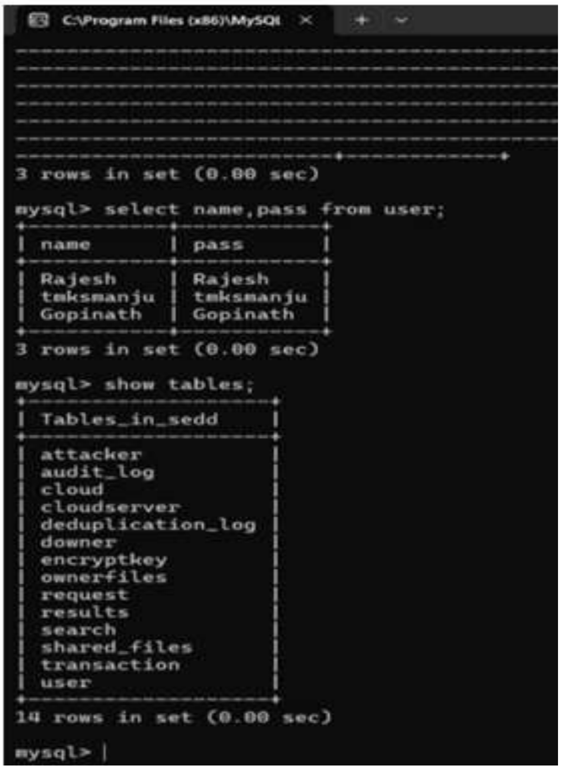
Name	Date modified	Type	Size
Drivers	[illegible]	File folder	
inetpub	[illegible]	File folder	
PerfLogs	[illegible]	File folder	
Program Files	[illegible]	File folder	
Program Files (x86)	[illegible]	File folder	
Users	[illegible]	File folder	
Windows	[illegible]	File folder	
Windows.old	[illegible]	File folder	
Database	[illegible]	SQL File	
logUploaderSettings	[illegible]	Configuration setti...	
logUploaderSettings_temp	[illegible]	Configuration setti...	

In above screen Databases application upload and now open brower and enter URL http://localhost:8080/SEDD/ and press enter key get below page



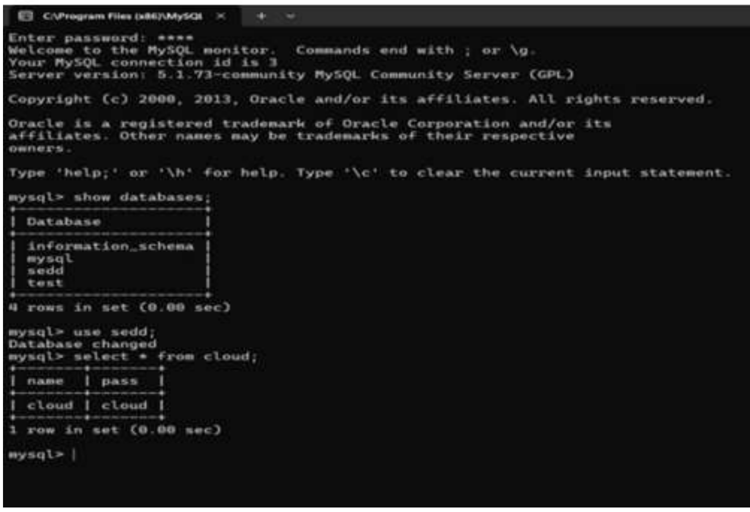
```
C:\Program Files (x86)\MySQL

mysql> show tables;
+---------------------+
| Tables_in_sedd      |
+---------------------+
| attacker            |
| audit_log           |
| cloud               |
| cloudserver         |
| deduplication_log   |
| downer              |
| encryptkey          |
| ownerfiles          |
| request             |
| results             |
| search              |
| shared_files        |
| transaction         |
| user                |
+---------------------+
14 rows in set (0.00 sec)

mysql> desc user;
+----------+----------+------+-----+---------+----------------+
| Field    | Type     | Null | Key | Default | Extra          |
+----------+----------+------+-----+---------+----------------+
| id       | int(11)  | NO   | PRI | NULL    | auto_increment |
| name     | text     | YES  |     | NULL    |                |
| pass     | text     | YES  |     | NULL    |                |
| email    | text     | YES  |     | NULL    |                |
| mobile   | text     | YES  |     | NULL    |                |
| addr     | text     | YES  |     | NULL    |                |
| dob      | text     | YES  |     | NULL    |                |
| gender   | text     | YES  |     | NULL    |                |
| pin      | text     | YES  |     | NULL    |                |
| location | text     | YES  |     | NULL    |                |
| imagess  | longblob | YES  |     | NULL    |                |
| status   | text     | YES  |     | NULL    |                |
+----------+----------+------+-----+---------+----------------+
12 rows in set (0.01 sec)

mysql>
```

The MySQL database shown belongs to the sedd schema, containing 14 tables related to secure cloud data storage and encryption, including attacker, cloud, encrypt key, requests, and transaction. The user table structure is displayed, consisting of fields such as id (primary key with auto-increment), name, pass, email, mobile, add, dob, gender, pin, location, images (stored as long blob), and status. This schema suggests a comprehensive system for managing user authentication, profile details, file encryption, and transaction tracking in a privacy-centric cloud environment.



```
---------------------------------------------------------------
---------------------------------------------------------------
---------------------------------------------------------------
---------------------------------------------------------------
---------------------------------------------------------------
---------------------------------------------------------------
--------------------------------+-------------+
3 rows in set (0.00 sec)

mysql> select name,pass from user;
+-------------+-------------+
| name        | pass        |
+-------------+-------------+
| Rajesh      | Rajesh      |
| tmksmanju   | tmksmanju   |
| Gopinath    | Gopinath    |
+-------------+-------------+
3 rows in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_sedd     |
+--------------------+
| attacker           |
| audit_log          |
| cloud              |
| cloudserver        |
| deduplication_log  |
| downer             |
| encryptkey         |
| ownerfiles         |
| request            |
| results            |
| search             |
| shared_files       |
| transaction        |
| user               |
+--------------------+
14 rows in set (0.00 sec)

mysql> |
```

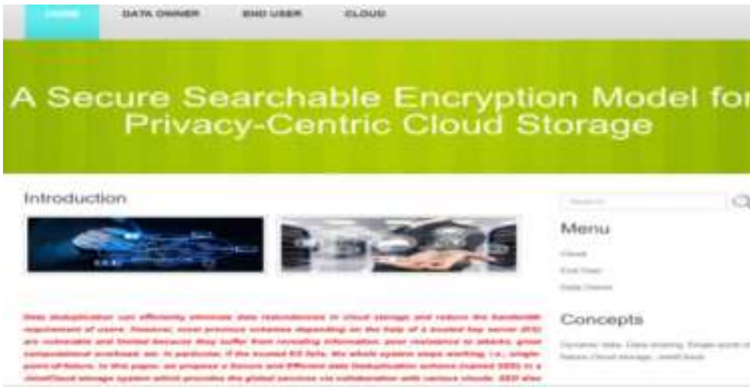
The screenshot displays a MySQL command-line interface connected to the sedd database. It first executes the query SELECT name, pass FROM user;, which returns three user accounts—Rajesh, tmksmanju, and Gopinath—along with their respective passwords stored in plain text. Following this, the SHOW TABLES; command is run, revealing that the database contains 14 tables: attacker, audit_log, cloud, cloud server, deduplication_log, downer, encryptkey, ownerfiles, request, results, search, shared_files, transaction, and user. This structure indicates a comprehensive system for managing



```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.1.73-community MySQL Community Server (GPL)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| sedd               |
| test               |
+--------------------+
4 rows in set (0.00 sec)

mysql> use sedd;
Database changed
mysql> select * from cloud;
+-------+-------+
| name  | pass  |
+-------+-------+
| cloud | cloud |
+-------+-------+
1 row in set (0.00 sec)

mysql> |
```

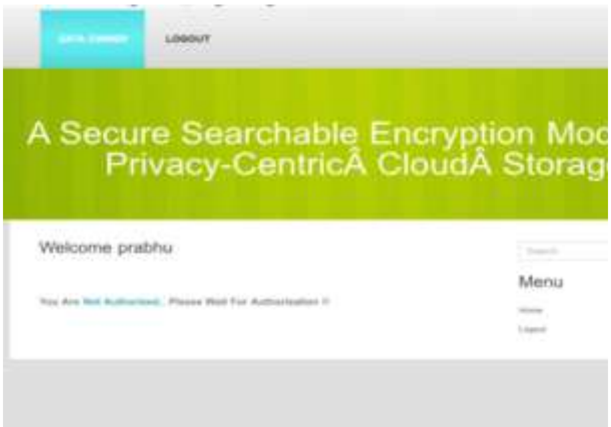
The MySQL terminal output shows that the secd database contains a table named cloud, which has one record with both name and pass fields set to "cloud". The query successfully retrieved the stored credentials from the database without any

errors.



HOME
DATA OWNER
END USER
CLOUD
A Secure Searchable Encryption Model for
Privacy-Centric Cloud Storage
Introduction
Menu
Concepts

This webpage presents a project titled "A Secure Searchable Encryption Model for Privacy-Centric Cloud Storage", which focuses on enhancing data privacy and reducing redundancy in cloud environments. It introduces an efficient data deduplication method integrated with secure encryption to optimize storage usage and ensure confidentiality in collaborative cloud systems.

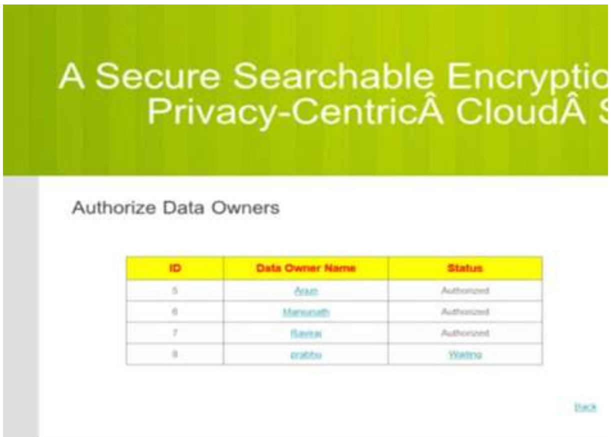


DATA OWNER
LOGOUT
A Secure Searchable Encryption Moc
Privacy-CentricÂ CloudÂ Storag
Menu
Home
Logout

Welcome prabhu

You Are Not Authorized , Please Wait For Authorization !!

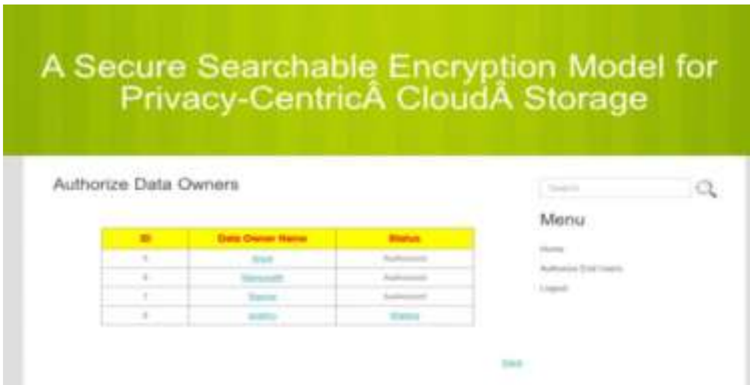
the authorization status for a logged-in data owner named Prabhu in the "Secure Searchable Encryption Model for Privacy-Centric Cloud Storage" system. Although the login was successful, the message indicates that the user is not yet authorized to access the platform's features. The system requires administrative approval before granting full access, ensuring that only verified and trusted users can manage or upload encrypted data. This security step helps maintain strict control over cloud storage resources and sensitive information.



Authorize Data Owners

ID	Data Owner Name	Status
5	Arjun	Authorized
6	Manjunath	Authorized
7	Ramesh	Authorized
8	prabhu	Waiting

the Data Owner Authorization section of the "Secure Searchable Encryption Model for Privacy-Centric Cloud Storage" system. It lists all registered data owners along with their authorization status, indicating which users have been granted access and which are still awaiting approval. In this example, Arjun, Manjunath, and Ramesh are authorized, while Prabhu is still in the waiting state. This feature allows the cloud administrator to manage access control, ensuring that only verified data owners can store and manage encrypted files in the system.

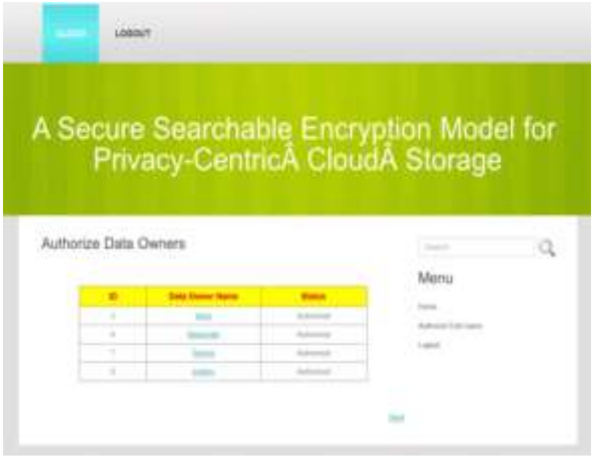


A Secure Searchable Encryption Model for
Privacy-CentricÂ CloudÂ Storage
Search
Menu
Home
Authorize End Users
Logout
Back

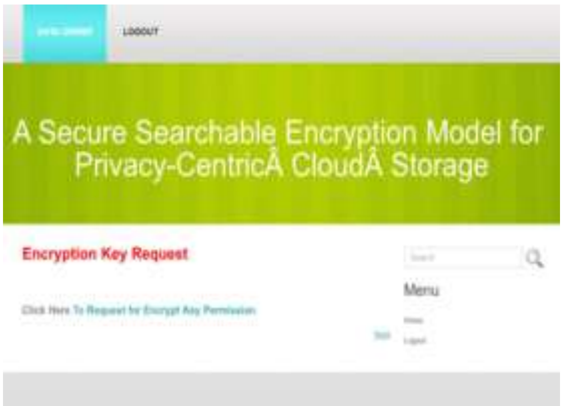
Authorize Data Owners

ID	Data Owner Name	Status
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]

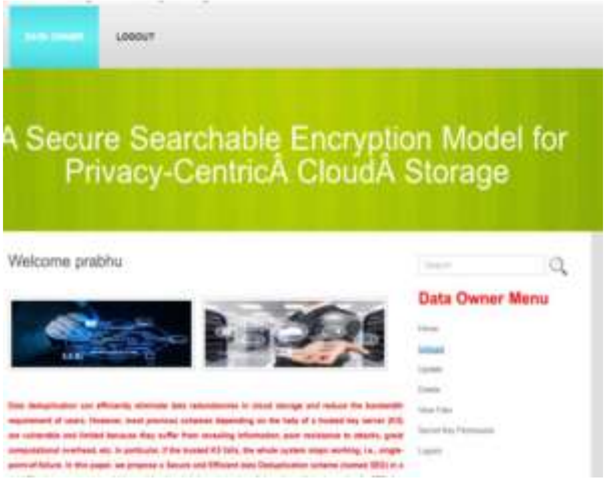
The Data Owner Authorization panel, where the cloud administrator can view and manage the approval status of registered data owners. It shows a list of users along with their IDs, names, and current status—either Authorized or Waiting. In this case, three data owners are already authorized, while one is still awaiting approval, ensuring controlled and secure access to the cloud storage system.


LOGOUT
A Secure Searchable Encryption Model for Privacy-CentricÂ CloudÂ Storage
Authorize Data Owners
ID
Data Owner Name
Status
Menu

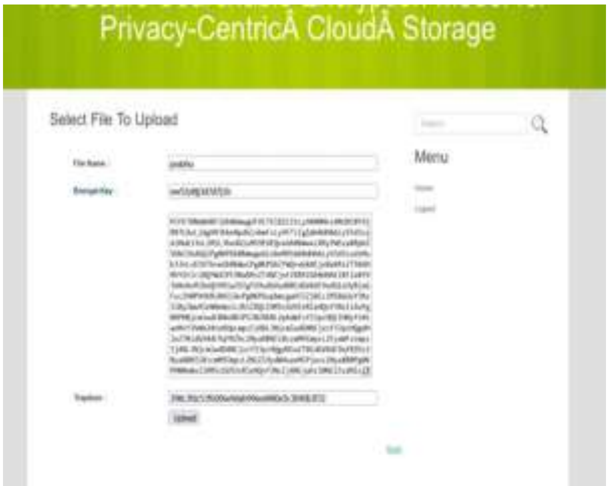
Data Owner Authorization interface of the cloud storage system, listing all registered data owners with their IDs, names, and authorization status. Here, all four data owners Arjun, Manjunath, Ramesh, and Prabhu are marked as Authorized. This indicates that they have been granted full access to securely store, manage, and encrypt their data within the privacy-centric cloud platform.


LOGOUT
A Secure Searchable Encryption Model for Privacy-CentricÂ CloudÂ Storage
Encryption Key Request
Click Here To Request for Encrypt Key Permission
Menu

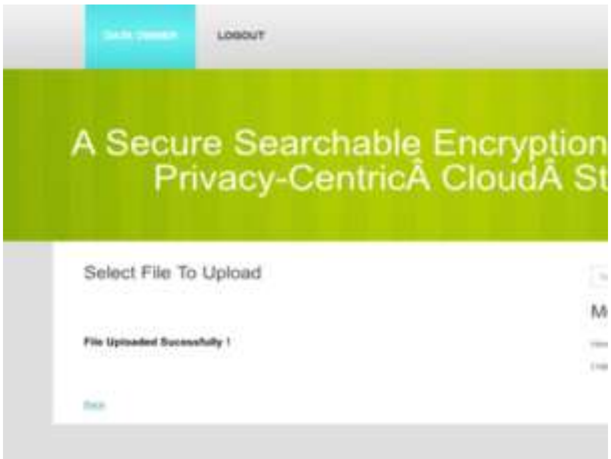
Data Owner Authorization section, confirming that all listed data owners have been granted access to the system. Each entry shows the owner’s ID, name, and status, with all marked as Authorized. This ensures that they can securely upload, manage, and encrypt their data in the privacy-focused cloud storage environment.


LOGOUT
A Secure Searchable Encryption Model for Privacy-CentricÂ CloudÂ Storage
Welcome prabhu
Data Owner Menu

The image shows a webpage titled "A Secure Searchable Encryption Model for Privacy-Centric Cloud Storage," welcoming a user named "prabhu." It includes a section on data deduplication and a "Data Owner Menu" with options like Home, Upload, Update, Delete, View Files, and Secret Key Permission Logout.



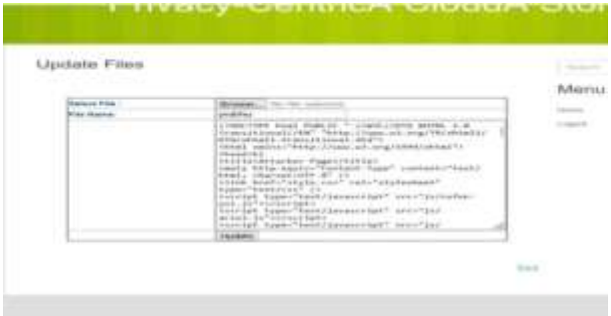
The user interface, titled "Select File To Upload," shows fields for a File Name (entered as "prabhu"), an Encrypt Key, and a Trapdoor. A large block of text, presumably the encrypted content, is visible below the "Encrypt Key" field. The presence of a "Trapdoor" field suggests the use of a searchable encryption scheme, where this value is used to securely search for the file without decrypting its contents.



Select File To Upload

File Uploaded Sucessfully !

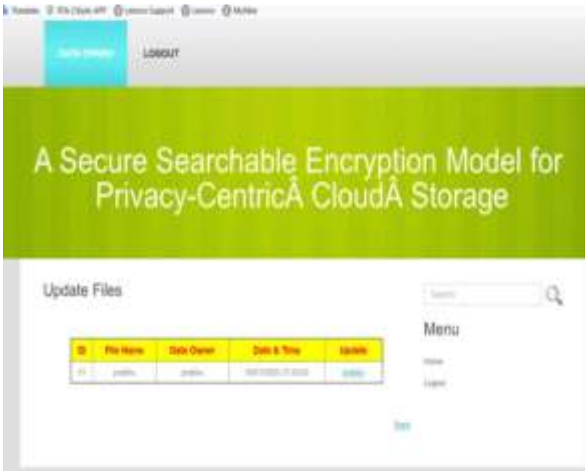
"A Secure Searchable Encryption Model for Privacy-Centric Cloud Storage" web application, hosted locally at http://localhost:8080/SEDO/Upload2.jsp, following a successful file upload. It displays the message "File Uploaded Successfully!" under the "Select File To Upload" heading, with a simple "Back" button below. The interface includes a "Menu" sidebar offering "Home" and "Logout" options, indicating completion of an upload process in this privacy-focused cloud storage prototype for the data owner role.



Update Files

Select File	Browse...
File Name	prabhu

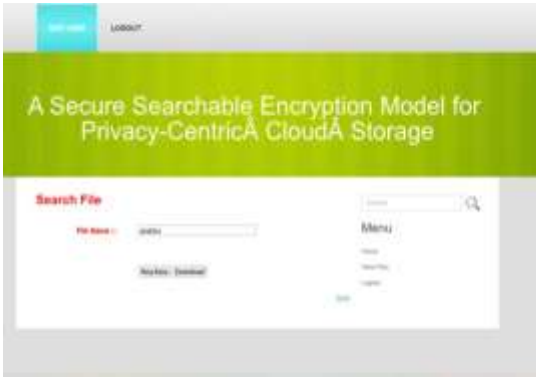
The image displays the user interface for managing files, possibly within a cloud storage or local file system application, indicated by elements like "Update Files" and a "Menu". This type of interface allows users to perform various operations such as uploading new files, organizing existing ones, and accessing different functionalities through the menu options. The visible table-like structure likely lists file details, including names, sizes, or modification dates, which are common features in file management systems designed to provide an organized overview of stored data.


DATA OWNER
LOGOUT
A Secure Searchable Encryption Model for
Privacy-CentricÂ CloudÂ Storage
Search
Menu
Home
Logout
Back

Update Files

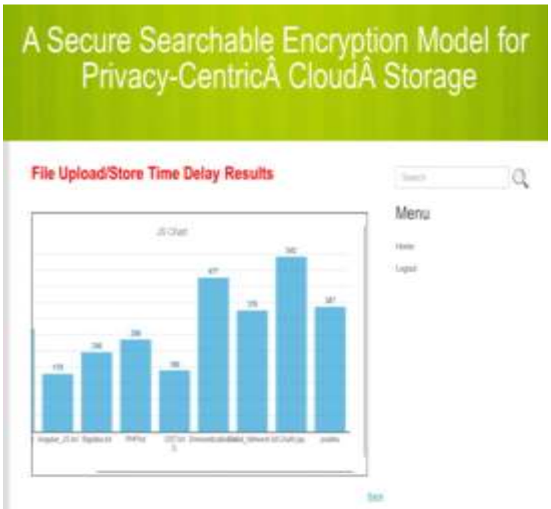
ID	File Name	Data Owner	Date & Time	Update
51	prabhu	prabhu	18/07/2025 21:30:03	Update

The screenshot shows the "Data Owner" interface of a system titled "A Secure Searchable Encryption Model for Privacy-Centric Cloud Storage". This module allows the data owner to manage and update stored files securely within the cloud environment. The "Update Files" section displays a table listing file details such as ID, File Name, Data Owner, Date & Time, and an Update link. In this instance, file ID 51 named prabhu is owned by the user prabhu and was last updated on 18/07/2025 21:30:03. The interface also includes a search bar for quick file lookup, a menu with navigation options like Home and Logout, and a clean, color-coded table design for better readability. This page plays a crucial role in enabling file modification while maintaining data privacy, security, and accountability within the encrypted cloud storage framework.

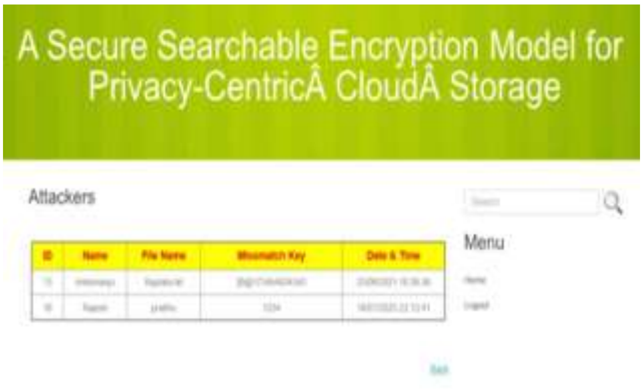


LOGOUT
A Secure Searchable Encryption Model for Privacy-CentricÂ CloudÂ Storage
Search File
File Name :
Req Keys Download
Menu

"A Secure Searchable Encryption Model for Privacy-Centric Cloud Storage." The user interface shows a search function where a user can enter a file name, specifically "prabhy," and then has options to "Req Keys" (Request Keys) or "Download." The application also features a simple menu with options for "Home," "View Files," and "Logout," suggesting a basic file management system focused on secure and private cloud storage.



The chart illustrates the file upload/store time delays for different files in a secure searchable encryption model for privacy-centric cloud storage. The chart illustrates the file upload/store time delays for different files in a secure searchable encryption model for privacy-centric cloud storage.



A Secure Searchable Encryption Model for Privacy-CentricÂ CloudÂ Storage
Attackers
Menu

ID	Name	File Name	Mismatch Key	Date & Time
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	prabhy	1234	[illegible]

The displayed page belongs to a cloud-based secure searchable encryption system that tracks unauthorized access attempts. It lists attackers who tried to download or decrypt files using incorrect keys. The table contains details such as attacker ID, name, targeted file name, the mismatched decryption key they used, and the exact date and time of the incident. This feature plays a crucial role in security monitoring by enabling system administrators to identify suspicious users, analyze intrusion attempts, and take preventive actions. Logging such activities enhances accountability and strengthens the overall privacy and integrity of the cloud storage environment.

The displayed interface shows a file download page in a secure file-sharing system. The page presents the contents of a file in a text area, allowing the user to review it before downloading. In this instance, the file appears to contain HTML and JavaScript code, possibly representing a web page.

CONCLUSION AND FUTURE ENHANCEMENT CONCLUSION In this article, we presented IM-DSSE, a new DSSE framework which offers very high privacy, efficient updates, low search latency simultaneously. Our constructions rely on a simple yet efficient incidence matrix data structure in combination with two hash tables that allow efficient and secure search and update operations. Our framework offers various DSSE constructions, which are specifically designed to meet the needs of cloud infrastructure and personal usage in different applications and environments. All of our schemes in IM-DSSE framework are proven to be secure and achieve the highest privacy among their counterparts. We conducted a detailed experimental analysis to evaluate the performance of our schemes on real Amazon EC2 cloud systems. Our results showed the high practicality of our framework, even when deployed on mobile devices with large datasets. We have released the full-fledged implementation of our framework for public use and analysis. Future Enchancement To improve the effectiveness and robustness of secure searchable encryption frameworks in privacy-critical cloud environments, several future enhancements can be considered. Integrating support for complex search queries, such as boolean, range, and fuzzy keyword searches, can greatly enhance usability and functionality. Incorporating machine learning-based query optimization may improve search efficiency and accuracy, especially over large encrypted datasets. Implementing search pattern and access pattern obfuscation techniques, such as Oblivious RAM (ORAM) or Private Information Retrieval (PIR), would further strengthen privacy by preventing leakage of user behavior. Enhancing the framework to support multi-user and role-based access control can allow for collaborative use while maintaining security boundaries. Additionally, integrating with blockchain technology for audit logging and tamper-proof records of search operations could add transparency and trust. Finally, optimizing performance for resource-constrained environments like mobile or edge devices would broaden the applicability of the system across diverse use

REFERENCES

[1] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: improved definitions and efficient constructions,” in Proc. 13th ACM Conf. Comput. Commun. security, ser. CCS ’06. ACM, 2006, pp. 79–88.

[2] E. Stefanov, C. Papamanthou, and E. Shi, “Practical dynamic searchable encryption with small leakage,” in 21st

Annu. Network and Distributed System Security Symp. — NDSS 2014. The Internet Soc., February 23-26, 2014.

[3] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in Proc. 2012 ACM Conf. Comput. Commun. security. New York, NY, USA: ACM, 2012, pp. 965–976.

[4] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. 2000 IEEE Symp. Security and Privacy, 2000, pp. 44–55.

[5] D. Cash, J. Jaeger, S. Jarecki, C. Jutla, H. Krawcyk, M.-C. Rosu, and M. Steiner, "Dynamic searchable encryption in very-large databases: Data structures and implementation," in 21th Annu. Network Distributed System Security Symp. — NDSS 2014. The Internet Soc., February 23-26, 2014.

[6] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," IEEE Trans. Parallel Distributed Syst., vol. 25, no. 1, pp. 222–233, 2014.

[7] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Verifiable privacy-preserving multi keyword text search in the cloud supporting similarity-based ranking," IEEE Trans. Parallel Distributed Syst., vol. 25, no. 11, pp. 3025–3035, 2014.

[8] S. Kamara and C. Papamanthou, "Parallel and dynamic searchable symmetric encryption," in Financial Cryptography and Data Security (FC), ser. Lecture Notes in Comput. Sci. Springer Berlin Heidelberg, 2013, vol. 7859, pp. 258–274.

[9] M. Naveed, M. Prabhakaran, and C. A. Gunter, "Dynamic searchable encryption via blind storage," in 35th IEEE Symp. Security Privacy, May 2014, pp. 48–62.

[10] F. Hahn and F. Kerschbaum, "Searchable encryption with secure and efficient updates," in Proc. 2014 ACM SIGSAC Conf. Comput. and Commun. Security. ACM, 2014, pp. 310–320.

[11] R. Bost, "Sophos – forward secure searchable encryption," in Proc. 2016 ACM Conf. Comput. Commun. Security. ACM, 2016.

[12] S. Kamara and T. Moataz, "Boolean searchable symmetric encryption with worst-case sub-linear complexity," EUROCRYPT 2017, 2017.

[13] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for boolean queries," in Advances in Cryptology, CRYPTO 2013, ser. Lecture Notes in Comput. Sci., vol. 8042, 2013, pp. 353–373.

[14] Z. Fu, X. Wu, C. Guan, X. Sun, and K. Ren, "Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement," IEEE Trans. Inform. Forensics Security, vol. 11, no. 12, pp. 2706–2716, 2016.

[15] Q. Wang, M. He, M. Du, S. S. Chow, R. W. Lai, and Q. Zou, "Searchable encryption over feature-rich data," IEEE

Trans. Dependable Secure Computing, 2016.

[16] Y. Zhang, J. Katz, and C. Papamanthou, “All your queries are belong to us: The power of file-injection attacks on searchable encryption,” in 25th USENIX Security ’16, Austin, TX, 2016, pp. 707–720.

[17] A. A. Yavuz and J. Guajardo, “Dynamic searchable symmetric encryption with minimal leakage and efficient updates on commodity hardware,” in Int. Conf. Selected Areas in Cryptography. Springer, 2015, pp. 241–259.

[18] P. Rizomiliotis and S. Gritzalis, “Oram based forward privacy preserving dynamic searchable symmetric encryption schemes,” in Proc. 2015 ACM Workshop Cloud Computing Security Workshop. ACM, 2015, pp. 65–76.

[19] E. Stefanov, M. Van Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu, and S. Devadas, “Path oram: an extremely simple oblivious ram protocol,” in Proc. 2013 ACM SIGSAC Conf. Comput. Commun. security. ACM, 2013, pp. 299–310.

[20] R. W. Lai and S. S. Chow, “Forward-secure searchable encryption on labeled bipartite graphs,” in Int. Conf. Appl. Cryptography Network Security. Springer, 2017, pp. 478–497.

[21] K. S. Kim, M. Kim, D. Lee, J. H. Park, and W.-H. Kim, “Forward secure dynamic searchable symmetric encryption with efficient updates,” in Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Security. ACM, 2017, pp. 1449–1463.

[22] X. Song, C. Dong, D. Yuan, Q. Xu, and M. Zhao, “Forward private searchable symmetric encryption with optimized i/o efficiency,” IEEE Trans. Dependable Secure Computing, 2018.

[23] M. Etemad, A. K¨upc¸ ¨ u, C. Papamanthou, and D. Evans, “Efficient dynamic searchable encryption with forward privacy,” Proc. Privacy Enhancing Technologies, vol. 2018, no. 1, pp. 5–20, 2018.

[24] R. Bost, B. Minaud, and O. Ohrimenko, “Forward and backward private searchable encryption from constrained cryptographic primitives,” in Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Security. ACM, 2017, pp. 1465–1482.

Mr. K. Jaya Krishna is an Associate Professor in the Department of Master of Computer Applications at QIS College of Engineering and Technology, Ongole, Andhra Pradesh. He earned his Master of Computer Applications (MCA) from Anna University, Chennai, and his M.Tech in Computer Science and Engineering (CSE) from Jawaharlal Nehru Technological University, Kakinada (JNTUK). With a strong research background, he has authored and co-authored over 90 research papers published in reputed peer-reviewed Scopus-indexed journals. He has also actively presented his work at various national and international conferences, with several of his publications appearing in IEEE-indexed proceedings. His research interests include Machine Learning, Artificial Intelligence, Cloud Computing, and Programming Languages. He is committed to advancing research and fostering innovation while mentoring students to excel in both academic and professional pursuits

.

Mr. POTHURI PRABHU KUMAR has received his MCA (Masters of Computer Applications) from QIS college of Engineering and Technology Vengamukkapalem(V), Ongole, Prakasam dist., Andhra Pradesh- 523272 affiliated to JNTUK in 2023-2025