VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI - 590 018

A Project Phase-1 Report

on

AI ENHANCED DUAL SERVER PUBLIC KEY ENCRYPTION FOR SECURE CLOUD

Submitted in partial fulfillment of the requirements for the final year degree in
Bachelor of Engineering in Computer Science and Engineering (Cyber Security)
of Visvesvaraya Technological University, Belagavi

Submitted by

Srushti S	1RN22CY039
B K Pramila	1RN23CY402

Under the Guidance of

Mr. Dhanraj
Assistant Professor
Dept. of CSE (Cyber Security)

ESTD : 2001
An Institute with a Difference

Department of CSE (Cyber Security)

RNS Institute of Technology

Affiliated to VTU, Recognized by GOK, Approved by AICTE,New Delhi
NACC 'A + Grade' Accredited, NBA Accredited (UG-CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr.Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080) 28611880, 28611881 URL:www.rnsit.ac.in

2024-2025

RN SHETTY TRUST®

RNS INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi
NACC 'A + Grade' Accredited, NBA Accredited (UG-CSE,ECE,ISE,EIE and EEE)
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098
Ph:(080)28611880,28611881 URL:www.rnsit.ac.in

DEPARTMENT OF CSE (Cyber Security)

ESTD : 2001
An Institute with a Difference

CERTIFICATE

Certified that the Project phase - 1 work entitled **PROJECT TITLE** has been successfully carried out at **RNSIT** by **Srushti S** bearing **1RN22CY039**, **B K Pramila** bearing **1RN23CY402**,bonafide students of **RNS Institute of Technology** in partial fulfillment of the requirements of final year degree in **Bachelor of Engineering in Computer Science and Engineering (Cyber Security) of Visvesvaraya Technological University, Belagavi** during academic year **2024-25** the Project phase - 1 report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Mr. Dhanraj
Assistant Professor
Dept. of CSE(CY), RNSIT

Dr. R Rajkumar
Project Coordinator
Assoc.Prof., Dept.of CSE(CY), RNSIT

Dr. Kiran P
Professor and Head
Dept. of CSE(CY) , RNSIT

Dr. Ramesh Babu H S
Principal
RNSIT

Acknowledgement

We express our profound thanks to the **Management of the RNS Institute of Technology** for fostering an environment that promotes innovation and academic excellence.

We would like place sincere gratitude to our beloved Director, **Dr. M K Venkatesha**, and our respected Principal, **Dr. Ramesh Babu H S** , for their constant encouragement and insightful support. Their guidance has been pivotal in keeping us motivated throughout this project work.

Our heartfelt appreciation goes to **Dr. Kiran P**, Professor and HoD of Computer Science and Engineering (Cyber Security), for his vital advice and constructive feedback, which significantly contributed to shaping this project work.

We also thank our Project Coordinator, **Dr. R Rajkumar**, Associate Professor, for his continuous monitoring, valuable suggestions and ensuring the process was deployed as per schedule.

We are deeply grateful to the project guide, **Mr. Dhanraj**, Assistant Professor, for his unwavering support, guidance, and valuable suggestions throughout the duration of this project. Lastly, our sincere thanks go to all the teaching and non-teaching staff members of the Computer Science and Engineering (Cyber Security) for their encouragement, cooperation, and support have been invaluable during this journey.

Warm Regards,

Srushti S (1RN22CY039)

B K Pramila (1RN23CY402)

Abstract

The increasing adoption of cloud computing has raised significant concerns regarding data security and privacy. To address these challenges, this project proposes an AI-enhanced dual server public key encryption system for secure cloud data storage and retrieval. The system leverages advanced cryptographic techniques, combining public-key encryption with dual-server architecture to enhance data confidentiality and integrity. The encryption process involves converting the plaintext data into a secure, hashed format using SHA-256 while generating an associated keyword hash for secure search functionality. The dual-server mechanism further enhances security by distributing encrypted data and keyword hashes across two separate servers, minimizing the risk of data compromise. The AI component intelligently manages secure data access and keyword-based searches, using trapdoor functions to verify encrypted keywords without revealing the actual content. The system is implemented using Flask for the web interface and Python for encryption, offering an interactive platform for secure data uploading and retrieval. Experimental results demonstrate the system's efficacy in maintaining data security, with efficient encryption and search operations. This approach provides a robust framework for secure cloud data management, combining AI-driven functionality with advanced encryption techniques.

Contents

List of Figures

Chapter 1

INTRODUCTION

Cloud computing has revolutionized data storage, offering scalability and cost-efficiency. However, it also introduces significant security and privacy challenges, particularly in protecting sensitive information from unauthorized access and breaches. Traditional encryption methods, especially single-server models, are vulnerable to various attacks, such as brute force and data interception.

To address these concerns, dual-server public key encryption has emerged as a more secure solution. By distributing encrypted data across two servers, the risk of a single-point failure is minimized. Additionally, integrating Artificial Intelligence (AI) enhances security further by enabling adaptive encryption and real-time breach detection.

This project proposes an AI-enhanced dual-server encryption system designed to securely store cloud data. By combining advanced encryption algorithms with AI-driven features like dynamic trapdoor mechanisms for keyword searching, the system improves both security and retrieval efficiency in cloud environments.

1.1 About the project

This project aims to develop an AI-enhanced dual-server public key encryption system designed to safeguard sensitive data in cloud environments. The proposed system leverages advanced encryption algorithms to securely transform plaintext data into ciphertext, incorporating keyword hashing for efficient and secure searching within the encrypted data. The dual-server architecture ensures that even if one server is compromised, the confidentiality and integrity of the data remain intact.

The system also incorporates a trapdoor mechanism, allowing for secure keyword-based searches without exposing the actual data. This feature reduces the risk of keyword guessing attacks and further strengthens the security of cloud data. Additionally, the integration of AI into the system makes it more adaptive, offering real-time monitoring and dynamic encryption, which enhances both security and performance.

To provide a user-friendly experience, the system will be implemented using Flask for the web interface and Python for the encryption processes. This combination of technologies will allow for secure management of cloud data in an easy-to-use platform.

By combining the dual-server encryption approach with AI-driven security features, this project addresses the key challenges associated with protecting sensitive cloud data. The system aims to offer an efficient, secure, and scalable solution for cloud storage, allowing organizations and individuals to manage their data safely in an increasingly complex security landscape.

1.2 Background and Motivation

The growing reliance on cloud services has intensified the need for enhanced data protection. As more data is moved to the cloud, concerns over confidentiality, integrity, and availability are rising. Data stored on remote servers is vulnerable to unauthorized access, breaches, and cyberattacks, prompting questions about the adequacy of current security methods.

Traditional encryption, while secure, often complicates efficient data retrieval, particularly with keyword-based searches. This inefficiency, along with rising threats like SQL injection and phishing attacks, calls for stronger, more resilient encryption models. Dual-server encryption, which distributes encrypted data across separate servers, mitigates the risks of single-point failures and protects data during searches.

Integrating AI into this system enhances security by detecting anomalies and adapting to evolving threats. AI-driven dynamic trapdoors further protect keyword searches without exposing sensitive data.

This project seeks to overcome the limitations of existing encryption methods by combining dual-server architecture with AI-powered encryption, offering a secure, efficient, and scalable solution for cloud data management and real-time searchability.

1.3 Existing System and its limitations

- **Lack of Multi-Layer Security:** Many cloud security systems rely on a single layer of protection, which can be bypassed if compromised. A more robust approach using multiple security layers (e.g., dual-server encryption) is often missing, leaving data vulnerable to attacks.
- **Poor Scalability:** Traditional encryption methods often struggle to scale effectively with increasing data volumes. As cloud data grows, the performance of these methods can degrade, slowing down data access and retrieval.
- **Slow Data Retrieval:** In current systems, retrieving encrypted data can be slow, especially when it involves searching for specific keywords. This inefficiency affects users who need quick access to important information.
- **Lack of Real-Time Adaptability:** Existing systems are often not designed to adapt quickly to new threats or changes in the cloud environment. This limits their ability to respond effectively to evolving security challenges.
- **Weak Protection Against Insider Threats:** Many cloud encryption systems do not adequately protect against insider threats. If a person with legitimate access to the system is compromised, the data can be easily exposed.

Chapter 2

LITERATURE SURVEY

2.1 Related Work

Paper1: Dual-Server Public-Key Authenticated Encryption with Keyword Search

The paper introduces [1] a new scheme called Dual-Server Public-Key Authenticated Encryption with Keyword Search (DPAEKS), which addresses the challenge of balancing protection against Insider Keyword Guessing Attacks (IKGA) and efficiency.DPAEKS utilizes two non-colluding servers to enhance security and ensures that the data owner is equipped with a pair of keys for authentication. The authors presented a concrete construction of DPAEKS, proved its security, and evaluated its performance.

Paper2: Oblivious Multi-Keyword Search for Secure Cloud Storage Service

The paper proposes OMKS protocols for encrypted data search, [2] offering three key innovations: (1) A basic oblivious keyword search protocol that reduces communication and storage overhead while ensuring database security and query privacy. (2) An extension to support both disjunctive and conjunctive multi-keyword searches, maintaining strong privacy without increasing overhead. (3) Analytical and experimental results showing that OMKS protocols are efficient, practical, and suitable for cloud environments, with stable performance even as query keywords increase.

Paper3: Time efficient privacy-preserving multi-keyword ranked search over encrypted cloud data

The paper introduces an efficient ranked keyword search (RSSE) [3] scheme for encrypted cloud data, using the OPSE crypto primitive for order-preserving mapping. Security analysis confirms its privacy, and experiments show its efficiency.

Paper4: Dual-Server Public-Key Authenticated Encryption with Keyword Search

This paper addresses data security in cloud storage, proposing a flexible distributed scheme with dynamic data support (update, delete, append) and data dependability through erasure-correcting codes. [4] The scheme uses homomorphic tokens for distributed verification of erasure-coded data, ensuring both storage correctness and error localization. It also supports third-party auditing, allowing users to delegate integrity checks. Security analysis and experiments show that the scheme is efficient and resilient to Byzantine failures, malicious data modification, and server collusion attacks.

Paper5: A Key Words Search for Secure Cloud Storage with Dual-Server Public-Key Encryption

This paper introduces Dual-Server Public Key Encryption with Keyword Search (DS-PEKS) to protect against keyword guessing attacks, a vulnerability in traditional PEKS. [5] It also presents a new Smooth Projective Hash Function (SPHF), which is used to build a generic DS-PEKS scheme. An efficient implementation of SPHF based on the Diffie-Hellman problem is demonstrated, providing an effective DS-PEKS scheme without the need for pairings.

Paper6: Dual-Server Public-Key Authenticated Encryption with Keyword Search

This paper explores forward security for public key searchable encryption, ensuring that newly added encrypted data cannot be searched using previously generated search tokens. [6] This feature is crucial for enhancing privacy and reducing information leakage in cloud storage. The study presents a solution to address this issue, which is essential for securing public key searchable encryption schemes in cloud environments.

Paper7: Secure keyword search using dual encryption in cloud computing

This paper presents a multiuser attribute-value database-based DSSE scheme for cloud-assisted eHealth systems. [7] Introduces a triple dictionary index structure and employs an identity server for secure multi-user search operations throughout the encrypted database. The scheme ensures L-adaptive security and forward security. Performance evaluations demonstrate its efficiency for practical use in eHealth applications.

Paper8: Efficient Data Retrieval Over Encrypted Attribute-Value Type Databases in Cloud-Assisted Ehealth Systems

This paper proposes a multiuser DSSE scheme for cloud-assisted eHealth systems, featuring a triple dictionary index structure and utilizing an identity server. This allows authorized users, such as researchers or medical staff, to search the entire encrypted database. [8] The scheme ensures L-adaptive security and forward security, with performance evaluations demonstrating its efficiency for practical use.

Paper9: Dual-Server Public-Key Encryption With Keyword Search for Secure Cloud Storage

This paper introduces Dual-Server Public Key Encryption with Keyword Search (DS-PEKS) [9] to prevent keyword guessing attacks, a vulnerability in traditional PEKS. It presents a new Smooth Projective Hash Function (SPHF), used to construct a generic DS-PEKS scheme. The paper also demonstrates an efficient implementation of SPHF based on the Diffie-Hellman problem, providing a DS-PEKS scheme without pairings.

2.2 Relevant recent paper's summary

In the era of cloud computing, ensuring both efficient and secure access to sensitive data has become a critical challenge. Searchable encryption (SE) techniques allow encrypted data to remain searchable without compromising confidentiality. Public-Key Encryption with Keyword Search (PEKS) is a significant approach within SE, enabling data users to retrieve encrypted files based on keywords. However, traditional PEKS schemes are highly susceptible to Inside Keyword Guessing Attacks (IKGA), where a malicious server can infer the queried keywords. To mitigate this issue, the paper introduces Dual-server Public-key Authenticated Encryption with Keyword Search (DPAEKS), a novel approach that employs two non-colluding servers and enforces authentication to defend against IKGA. Unlike earlier schemes, DPAEKS requires both the public keys of the servers and a shared key between the data owner and user, thereby ensuring only authenticated users can perform searches. Additionally, the scheme is constructed without the computationally expensive bilinear pairings and is proven secure under the Decisional Diffie-Hellman (DDH) assumption. Experiments on real-world datasets show that DPAEKS offers strong security and high efficiency, making it viable for practical deployment in cloud storage systems.

2.3 Summary

The DPAEKS framework uses dual-server encryption to prevent insider keyword guessing attacks, improving both security and efficiency for encrypted data retrieval. The OMKS protocols offer oblivious multi-keyword search with minimal overhead, ensuring strong privacy and making them suitable for cloud environments. A new ranked searchable encryption scheme based on order-preserving encryption (OPSE) enables efficient ranked search over encrypted data. In a distributed cloud storage scheme, data integrity and availability are assured using erasure codes and homomorphic tokens, with support for third-party auditing. DS-PEKS introduces protection against keyword guessing attacks through a dual-server architecture and a Smooth Projective Hash Function (SPHF), implemented without pairings. Forward security in public key searchable encryption ensures that new data cannot be searched with outdated tokens, reducing privacy risks. A multiuser DSSE scheme for eHealth enables secure, searchable access to encrypted medical records using a triple dictionary index and identity server. The DS-PEKS framework (repeated) enhances security by utilizing SPHF based on the Diffie-Hellman problem to prevent insider keyword guessing. Lastly, a multiuser searchable encryption system for healthcare settings guarantees efficiency, forward security, and practicality through comprehensive performance evaluations.

Chapter 3

PROBLEM STATEMENT

Cloud storage offers scalability and convenience but poses serious security and privacy risks, especially during keyword-based searches. Traditional single-server encryption methods are vulnerable to breaches and inefficient for secure search. This project addresses these challenges by developing an AI-enhanced dual-server encryption system that ensures data confidentiality while enabling fast, secure keyword retrieval.

3.1 Objectives

This project aims to develop a secure, intelligent cloud storage system that utilizes AI and dual-server encryption to ensure data confidentiality and enable efficient, privacy-preserving keyword search.

- To develop an AI-enhanced dual server public key encryption system for secure cloud data storage and retrieval.
- To implement an adaptive encryption technique that utilizes dual server architecture to minimize single-point failures.
- To integrate dynamic trapdoor mechanisms for secure and efficient keyword-based searches without compromising data confidentiality.
- To evaluate the system's performance through experimental analysis, focusing on encryption efficiency, data retrieval speed, and security robustness.

Chapter 4

SYSTEM ARCHITECTURE

4.1 System Model Description

The system consists of four main entities:

- DO (Data Owner) – The entity that uploads encrypted data along with searchable indexes (keywords).
- AS (Authorization Server) – One of the two cloud servers that handles part of the data storage and keyword search processing.
- TS (Test Server) – The second server that works collaboratively with the AS to ensure secure and efficient keyword search while protecting user privacy.
- DR (Data Receiver/Requester) – The authorized user who wants to search for specific data using encrypted queries.

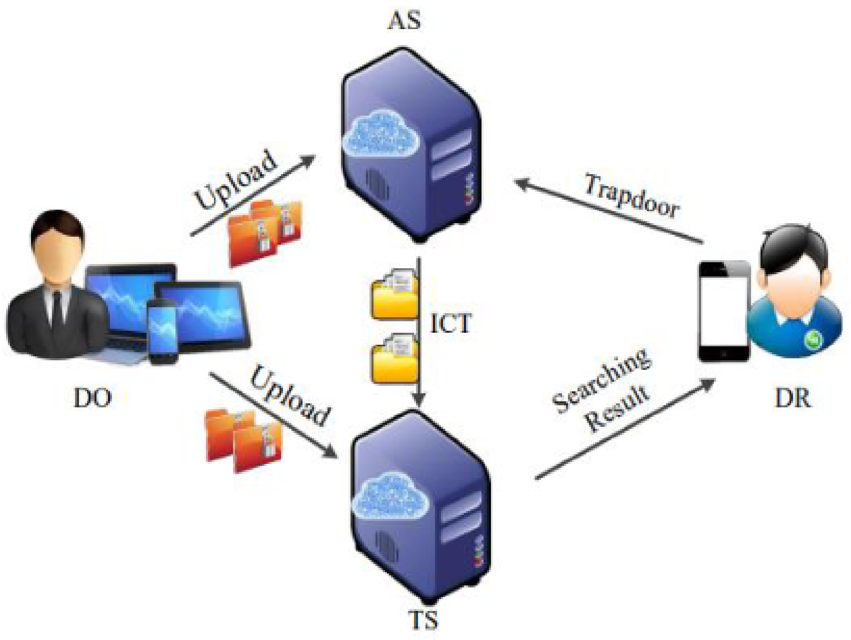


Figure 4.1: System Model

Flow of Operations:

- Upload Phase: The Data Owner (DO) encrypts data and generates a secure searchable index (ICT). The encrypted data and ICT are uploaded to both AS and TS, distributing the storage and preventing full control by a single server.
- Trapdoor Generation: The Data Receiver (DR), who wants to search for a specific keyword, generates a Trapdoor (an encrypted search token) using the public parameters and their private key.
- Search Operation: The Trapdoor is sent to the AS, which uses it to initiate a search on the encrypted index. AS collaborates with TS using secure protocols without revealing the keyword or actual data.
- Result Retrieval: The Search Result is computed and returned to the DR. Since both servers (AS and TS) participated in the process, the system benefits from improved security and resistance to leakage (like keyword guessing attacks).

Why Dual Server?

The dual-server architecture prevents either the AS or TS from having full control or knowledge of the data and search terms. This division of responsibilities enhances security by:Minimizing leakage of search queries and access patterns.Avoiding reliance on a single point of trust.Enabling privacy-preserving keyword searches.

4.2 Cloud Functionality Requirements

We are implementing our project with real time cloud and using AWS for cloud storage
Why AWS for Cloud Storage: AWS offers highly scalable, real-time cloud storage solutions with global availability, ensuring low latency and high reliability for our applications. Its robust security framework and compliance certifications make it ideal for handling sensitive data. Additionally, AWS provides seamless integration with numerous services that enhance data processing and management.

The system is designed to enable users to upload plaintext data through a user-friendly web interface. Once the data is uploaded, it undergoes encryption before being securely stored on a cloud server, ensuring data confidentiality. Each keyword entered by the user is processed using a cryptographic hash function and transformed into a trapdoor, a secure and private representation of the keyword. The cloud server is equipped with the capability to perform searches over this encrypted data using the trapdoors, without ever revealing the original keywords, thereby preserving user privacy.
An assistant server plays a critical role in this architecture by generating and verifying intermediate ciphertexts that facilitate secure and efficient search matching. When a user initiates a search, the system verifies the correspondence between the trapdoor and stored encrypted data, and then returns a simple match or no-match response. To maintain the integrity and security of data during communication, all exchanges between different modules of the system are conducted over HTTP or other secure protocols.

4.3 System Security and Performance

We are implementing our project with SHA-256 encryption algorithm.
Why SHA-256 for Encryption: SHA-256 is a widely trusted cryptographic hash function that produces a fixed-length, 256-bit hash, ensuring consistent and secure data integrity verification. Its robustness against collision and pre-image attacks makes it a preferred choice over older algorithms

like MD5 or SHA-1. The algorithm's efficiency and widespread industry adoption further justify its use in our project.

The system must ensure end-to-end confidentiality by employing SHA-256 hashing for secure keyword representation and AES-CBC encryption for protecting the uploaded plaintext data. These cryptographic measures are critical in preventing keyword leakage and ensuring that even if data is intercepted or accessed directly from the cloud, it remains unreadable in its encrypted form. The hashing of keywords into trapdoors must be designed in such a way that no information about the original keywords can be inferred. To provide a responsive user experience, the system should be capable of processing search queries and returning results within 1–2 seconds under normal operating conditions. The user interface should be designed to be intuitive and straightforward, ensuring accessibility from any standard web browser without the need for specialized software. Furthermore, the system architecture must follow a modular and maintainable design, making it easier to incorporate future enhancements, such as AI-based features or upgraded encryption algorithms. Looking ahead, future updates may introduce parallel search execution to boost system performance and reduce query response times even further.

4.4 User Interaction and Accessibility

- Users can upload data and enter keywords through a user-friendly web page.
- The system will automatically handle encryption and keyword processing in the background.Users will receive clear feedback if a keyword match is found or not.
- No technical knowledge is required from the user to operate the system.The interface should work smoothly on common web browsers without additional setup.The system should protect user data during upload, storage, and search.

Chapter 5

IMPLEMENTATION

5.1 Dataset Collection

In this project, the dataset is dynamically collected through the web-based application interface, where users upload textual data along with a related keyword. Instead of using a predefined or static dataset, the system allows users to generate and store their own content in real-time. Each uploaded file is immediately processed—its content is encrypted using secure algorithms, and the keyword is hashed using SHA-256 to generate a trapdoor for secure keyword-based search. This dynamic data collection approach ensures flexibility and simulates a real-world cloud environment where sensitive information is securely stored and accessed through privacy-preserving search mechanisms. The use of AWS cloud storage enables real-time scalability, high availability, and secure access to encrypted data.

5.2 Software Requirements

Libraries Used:

Flask – For creating the web interface and handling routes (app.py).

hashlib – For SHA-256 hashing operations (encryption.py).

Algorithm Used:

SHA-256 – Used for: Encrypting uploaded data (encrypt data()).

Hashing search keywords (hash keyword()).

Generating trapdoors (generate trapdoor()).

Trapdoor-Based Keyword Search – Implements secure search by:

Creating an intermediate ciphertext from encrypted keywords and trapdoors (assistant server.py).

Verifying matches using intermediate values (test server.py).

5.3 Algorithms and Methods

Algorithm 1: Data and Keyword Encryption

This algorithm handles encryption of file data and the associated keyword. The file content is encrypted using SHA-256 hashing to ensure security and data integrity. Similarly, the keyword is also hashed to allow for secure and private keyword matching. An additional trapdoor is generated by hashing a modified form of the keyword, which will be used later for verification.

```
 1: procedure EncryptData(data)
 2:     encrypted ← SHA-256(data)
 3:     return encrypted
 4: end procedure
 5: procedure HashKeyword(keyword)
 6:     hashed_keyword ← SHA-256(keyword)
 7:     return hashed_keyword
 8: end procedure
 9: procedure GenerateTrapdoor(keyword)
10:     trapdoor ← SHA-256("trap" + keyword)
11:     return trapdoor
12: end procedure
```

Algorithm 2: Web Application and Control Logic

This algorithm defines the control flow of the main web application using Flask. It includes routes for uploading files and searching for keywords. The upload route encrypts the file and keyword, while the search route checks for a match between the stored hashed keyword and the user-provided input. It acts as the frontend-controller for interaction between users and the backend logic.

```
 1: Initialize Flask app and empty cloud storage
 2: Route /index:
 3:     Render index.html with encrypted data, keyword hash, last query, and result
 4: Route /upload:
 5:     Get file data and keyword from form
 6:     Encrypt file data using ENCRYPTDATA
 7:     Hash keyword using HASHKEYWORD
 8:     Store encrypted data and hashed keyword
 9: Route /search:
10:     Get keyword from form and hash it
11:     Compare hash with stored keyword hash
12:     Update result as ”Match Found” or ”No Match”
```

Algorithm 3: Intermediate Ciphertext Generation

This algorithm is responsible for generating the Intermediate Cipher Text (ICT) used in the dual-server architecture. It extracts the first 16 characters from the encrypted keyword and the last 16 from the trapdoor. These parts are concatenated to form the ICT, which is used for secure verification without exposing sensitive information.

Algorithm 1 Generate Intermediate Ciphertext

```
1: procedure CREATEINTERMEDIATECIPHERTEXT($encrypted\_keyword$, $trapdoor$)
2:     $prefix \leftarrow encrypted\_keyword[0:16]$
3:     $suffix \leftarrow trapdoor[-16:]$
4:     $intermediate \leftarrow prefix + suffix$
5:     return $intermediate$
6: end procedure
```

Algorithm 4: Match Verification Logic

This algorithm checks whether the Intermediate Cipher Text is valid by reconstructing it from the stored encrypted keyword and trapdoor. If the reconstructed ICT matches the received one, a successful match is confirmed. This ensures that both the keyword and the trapdoor were generated correctly and belong together, enabling secure verification.

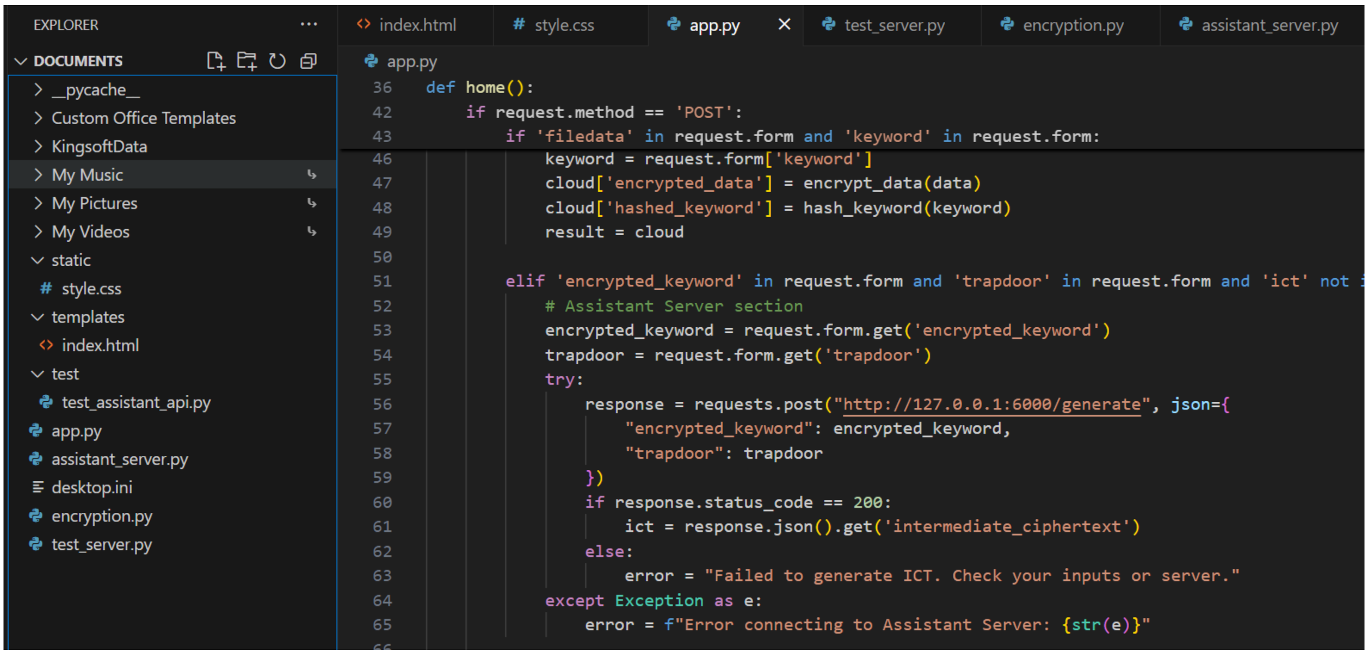
Algorithm 2 Verify Intermediate Ciphertext

```
procedure VerifyIntermediate($intermediate$, $encrypted\_keyword$, $trapdoor$)
    $expected \leftarrow encrypted\_keyword[0:16] + trapdoor[-16:]$
    if $intermediate = expected$ then
        return "Match confirmed"
    else
        return "Mismatch"
    end if
end procedure
```

Chapter 6

TESTING AND RESULTS

This page shows the files that we are created in the Visual Studio there are index for web page,assistant-server,test-server,style like files which are working according to there functioning.

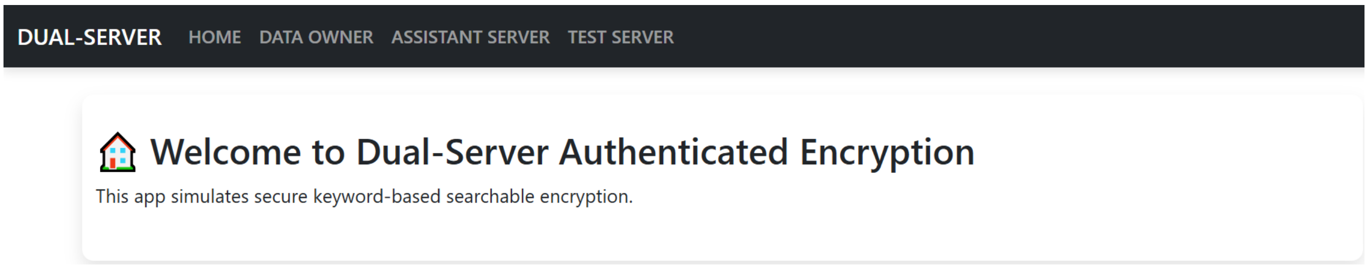


```python
def home():
    if request.method == 'POST':
        if 'filedata' in request.form and 'keyword' in request.form:
            keyword = request.form['keyword']
            cloud['encrypted_data'] = encrypt_data(data)
            cloud['hashed_keyword'] = hash_keyword(keyword)
            result = cloud

        elif 'encrypted_keyword' in request.form and 'trapdoor' in request.form and 'ict' not
            # Assistant Server section
            encrypted_keyword = request.form.get('encrypted_keyword')
            trapdoor = request.form.get('trapdoor')
            try:
                response = requests.post("http://127.0.0.1:6000/generate", json={
                    "encrypted_keyword": encrypted_keyword,
                    "trapdoor": trapdoor
                })
                if response.status_code == 200:
                    ict = response.json().get('intermediate_ciphertext')
                else:
                    error = "Failed to generate ICT. Check your inputs or server."
            except Exception as e:
                error = f"Error connecting to Assistant Server: {str(e)}"
```

Figure 6.1: Visual Studio Page

The Home Page of the Dual-Server Web App serves as a welcoming interface introducing users to the project. It briefly explains the purpose of the app, which is to simulate secure keyword-based searchable encryption using a dual-server approach. Users can navigate to different panels like Data Owner, Data Receiver, Test Server, and Assistant Server. The clean layout with clear navigation ensures ease of use and quick access to each functionality.



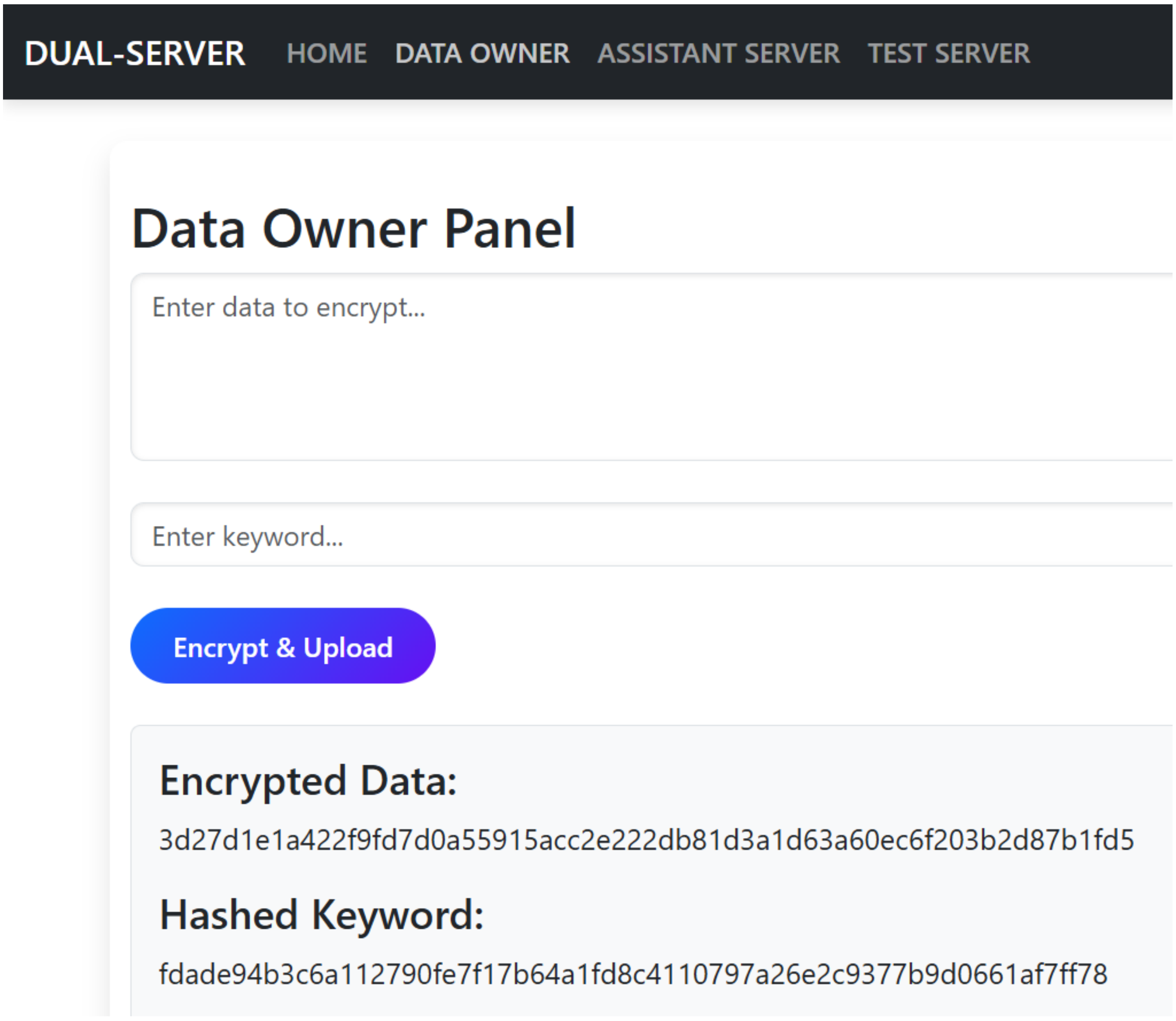
Welcome to Dual-Server Authenticated Encryption

This app simulates secure keyword-based searchable encryption.

Figure 6.2: Home page

- Data Owner Panel This section allows the user (data owner) to input sensitive data and a keyword. The data is encrypted, and the keyword is hashed before being stored, simulating secure data upload.
- Data Receiver Panel This area is reserved for the future implementation of the receiver's role. It will allow authorized users to retrieve and decrypt data using search tokens.
- Test Server Panel This panel is used to validate whether the intermediate ciphertext (ICT) is generated correctly. It checks if the server logic for verification is functioning as intended.
- Assistant Server Panel Here, the Assistant Server creates an intermediate ciphertext using the encrypted keyword and a trapdoor. This helps simulate cooperative search without exposing sensitive values.

This interface allows the user to upload a file and enter an associated keyword. On clicking Upload, the system encrypts the file content and stores it securely on the cloud server (AWS or simulated cloud).The keyword is hashed for secure keyword search functionality.



Data Owner Panel

Encrypted Data:

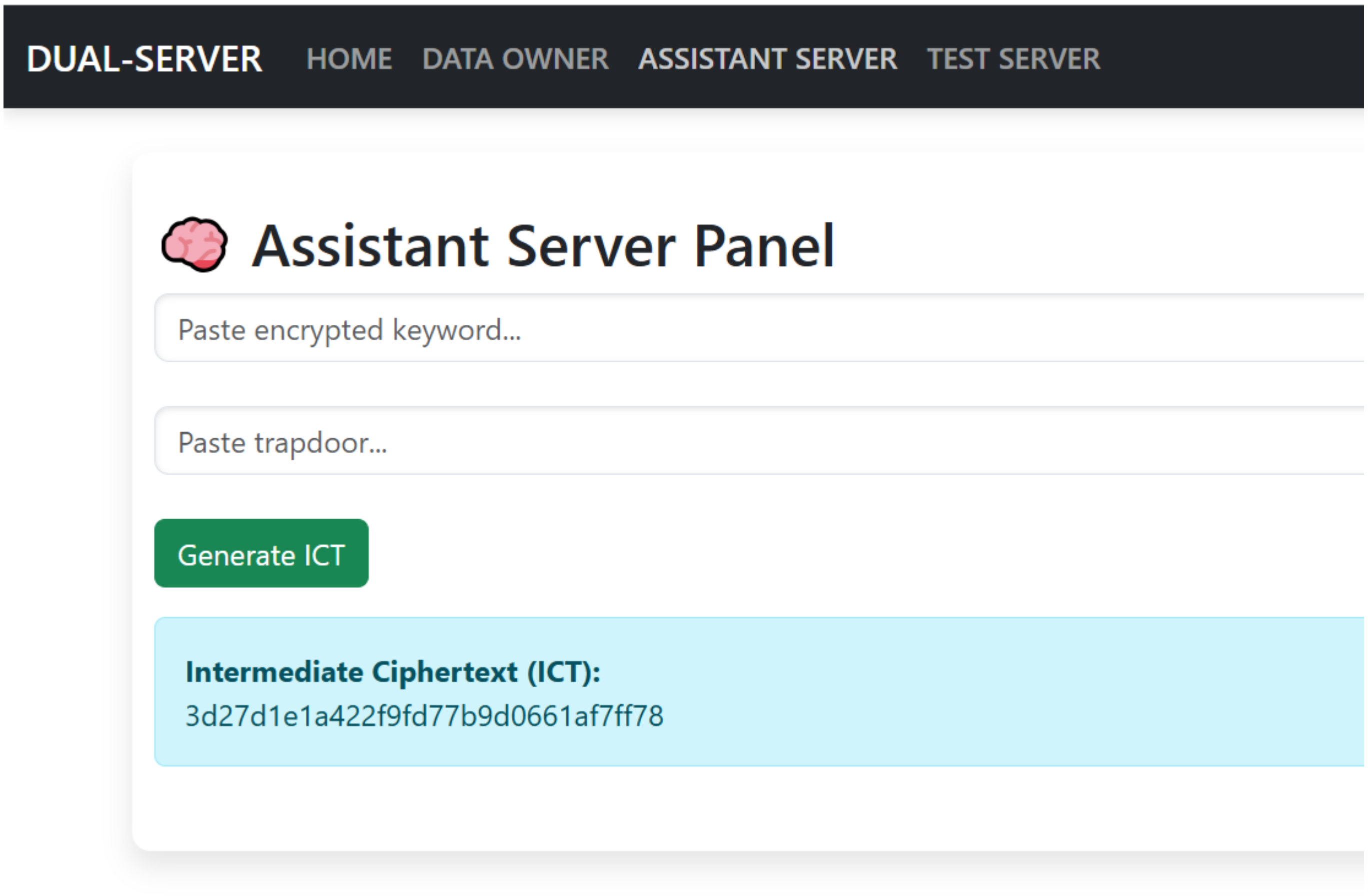
3d27d1e1a422f9fd7d0a55915acc2e222db81d3a1d63a60ec6f203b2d87b1fd5

Hashed Keyword:

fdade94b3c6a112790fe7f17b64a1fd8c4110797a26e2c9377b9d0661af7ff78

Figure 6.3: Data uploading page

This page allows the assistant server to log in securely. The assistant server is responsible for handling trapdoor-based search verification by generating or validating intermediate ciphertext, supporting the dual-server architecture.
This process ensures that the actual keyword remains hidden while still allowing secure and efficient search operations on encrypted data. The Assistant Server adds a layer of security by preventing any single server from having complete access to sensitive information.



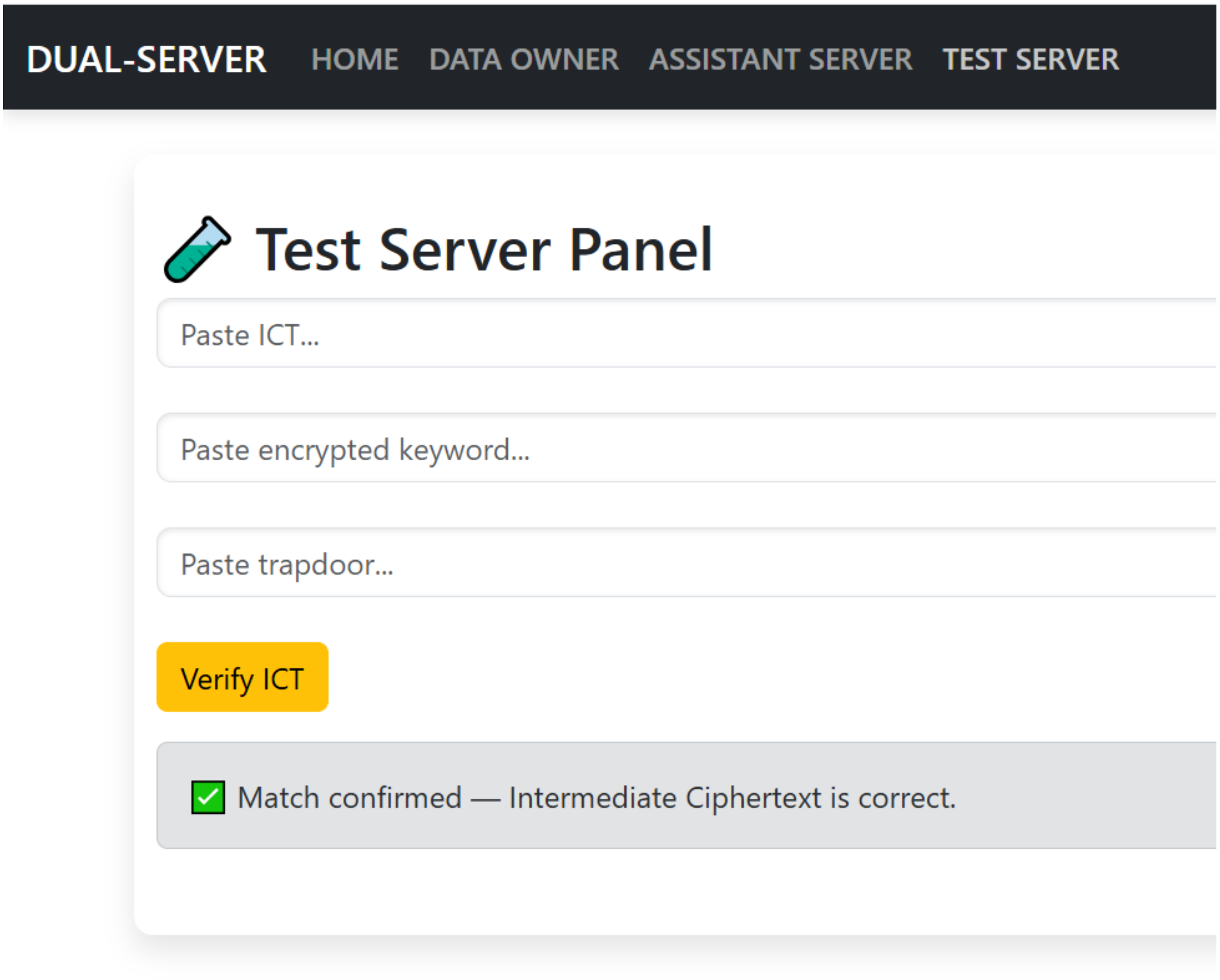
Assistant Server Panel

Intermediate Ciphertext (ICT):
3d27d1e1a422f9fd77b9d0661af7ff78

Figure 6.4: Assistant Server Login

.

This pop-up indicates successful login into the system. After logging in, the user can perform actions like uploading files, searching with keywords, or verifying results through the dual-server architecture.



Test Server Panel

Paste ICT...

Paste encrypted keyword...

Paste trapdoor...

✅ Match confirmed — Intermediate Ciphertext is correct.

Figure 6.5: ICT confirmation

Chapter 7

CONCLUSION AND FUTURE-ENHANCEMENT

7.1 Conclusion

The project successfully implements a secure cloud-based storage system that ensures confidentiality of user data through encryption. By using SHA-256 hashing, the system prevents direct access to both data and keywords, preserving privacy even during search operations. The concept of trapdoor generation and intermediate ciphertext supports a dual-server architecture, enhancing the security by separating data storage and search verification roles. Users can upload, store, and search encrypted files via a simple and accessible web interface, simulating a real-time cloud environment using platforms like AWS. The system ensures that keyword-based searches can be performed without revealing the actual keyword or decrypting the data, maintaining both functionality and privacy. This project demonstrates a practical solution for secure data retrieval in cloud environments, which can be scaled further with AI enhancements and more advanced encryption like AES-CBC.

7.2 Future Enhancement

In the next phase of this project, we plan to integrate Artificial Intelligence into the searchable encryption framework. The goal is to move beyond exact keyword matching and introduce intelligent, context-aware search capabilities.

By leveraging Natural Language Processing (NLP), users can input questions or vague queries instead of exact keywords. The AI model will interpret the intent and match it with the most relevant encrypted documents.This enhancement would greatly improve user experience by reducing dependency on exact keyword recall. For example, searching for “student grades” can also fetch results tagged as “academic scores” or “performance sheets.”

We aim to train a lightweight AI model that runs client-side or on a secure server, ensuring data privacy is preserved during query processing. The trapdoor mechanism will still be retained, but improved with semantic-level search embedding. We are also considering using federated learning to train models securely without exposing raw data to external systems. AI-based ranking will further improve the order of search results.

This upgrade will turn the current system into an intelligent, privacy-preserving searchable encryption engine. It bridges the gap between security and usability, enabling smarter access to encrypted information.The implementation will begin in the 7th semester as part of our advanced project work. Initial results will be tested using dummy data and gradually scaled for real-world use. Our aim is to demonstrate a secure, AI-powered, dual-server system by the end of the 8th semester.

References

[1] B. Chen, L. Wu, S. Zeadally and D. He, ”Dual-Server Public-Key Authenticated Encryption with Keyword Search,” in IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 322-333, 1 Jan.-March 2022, doi: 10.1109/TCC.2019.2945714.

[2] R. Zhang, R. Xue, L. Liu and L. Zheng, ”Oblivious Multi-Keyword Search for Secure Cloud Storage Service,” 2017 IEEE International Conference on Web Services (ICWS), Honolulu, HI, USA, 2017, pp. 269-276, doi: 10.1109/ICWS.2017.42.

[3] A. B. Jivane, ”Time efficient privacy-preserving multi-keyword ranked search over encrypted cloud data,” 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, India, 2017, pp. 497-503, doi: 10.1109/ICPCSI.2017.8392345.

[4] B. Chen, L. Wu, S. Zeadally and D. He, ”Dual-Server Public-Key Authenticated Encryption with Keyword Search,” in IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 322-333, 1 Jan.-March 2022, doi: 10.1109/TCC.2019.2945714.

[5] article:reddy2017key,title:A Key Words Search for Secure Cloud Storage with Dual-Server Public-Key Encryption author: REDDY, MANDAPATI NARENDRA and ROHITH, P and SREE, P ARUNA,2017

[6] Chen, L. Wu, S. Zeadally and D. He, ”Dual-Server Public-Key Authenticated Encryption with Keyword Search,” in IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 322-333, 1 Jan.-March 2022, doi: 10.1109/TCC.2019.2945714.

[7] Tariq, H., Agarwal, P. (2020). Secure keyword search using dual encryption in cloud computing. International Journal of Information Technology, 12(4), 1063-1072.

[8] S. Li et al., ”Efficient Data Retrieval Over Encrypted Attribute-Value Type Databases in Cloud-Assisted Ehealth Systems,” in IEEE Systems Journal, vol. 16, no. 2, pp. 3096-3107, June 2022, doi: 10.1109/JSYST.2021.3073169.

[9] R. Chen, Y. Mu, G. Yang, F. Guo and X. Wang, ”Dual-Server Public-Key Encryption With Keyword Search for Secure Cloud Storage,” in IEEE Transactions on Information Forensics and Security, vol. 11, no. 4, pp. 789-798, April 2016, doi: 10.1109/TIFS.2015.2510822.