JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Efficient Dual-Server Public-Key Authenticated Encryption with Keyword Search for Privacy-Preserving Cloud Storage

Mrs. Shivam Patkar[1] Rajneesh Pachouri[2] Mr. Anurag Jain[3],
Research Scholar, Assistant Professor
Department of Computer Science and Engineering
Adina Institute of Science & Technology, Sagar, India

Abstract : This study presents a Dual-Server Public-Key Authenticated Encryption with Keyword Search (DPAEKS) system designed to provide secure and efficient retrieval of sensitive data stored in cloud environments. The proposed model addresses the limitations of traditional public-key encryption with keyword search (PEKS) schemes, which are vulnerable to inside keyword guessing attacks and often lack strong user authentication and practical performance. Building on recent advances in authenticated searchable encryption and dual-server architectures, the scheme introduces two non-colluding servers—Assistant Server and Test Server—that cooperatively execute the keyword search Test operation, ensuring that neither server alone can complete the search or infer the queried keyword. Data owners encrypt documents and generate authenticated keyword cipher texts using their own keys, the public keys of both servers, and a shared key with authorized data receivers, while data receivers generate authenticated trapdoors for their search queries. The dual-server design with intermediate cipher texts enhances resistance to inside keyword guessing attacks, enforces strict access control, and maintains privacy of user interests. Experimental evaluation comparing the proposed DPAEKS system with traditional PEKS and single-server PAEKS shows reduced encryption, trapdoor generation, and search times, with only a moderate increase in communication overhead due to server cooperation. These results demonstrate that the proposed DPAEKS scheme offers a practical and robust solution for privacy-preserving keyword search over encrypted cloud data, achieving a favorable balance between strong security guarantees and real-world efficiency.

IndexTerms **-Dual-server architecture, public-key authenticated encryption, keyword search, searchable encryption, cloud data security, inside keyword guessing attack**

I. INTRODUCTION

Cloud computing has become an essential platform for storing and sharing large volumes of data because it offers on-demand access, scalability, and reduced infrastructure costs. When sensitive information such as medical records, financial data, or personal documents is outsourced to third-party cloud providers, however, data owners lose direct control over storage and management. Encrypting data before outsourcing is a fundamental way to preserve confidentiality, but it also makes traditional plaintext search impossible and significantly reduces the usability of cloud-hosted data. To overcome this limitation, searchable encryption techniques have been introduced, enabling keyword-based search over encrypted data without revealing document content to the server.

Public-Key Encryption with Keyword Search (PEKS) is an important searchable encryption paradigm that allows a data owner to encrypt documents along with keywords and enables an authorized data receiver to generate trapdoors for searching relevant cipher texts. Despite its conceptual elegance, many conventional PEKS schemes suffer from serious security weaknesses in practical cloud settings. A particularly critical issue is the inside keyword guessing attack (IKGA), where a curious or malicious cloud server uses its knowledge of trapdoors, cipher texts, and the typically small keyword space to iteratively test and recover the actual keywords queried by users. This attack exposes sensitive information about user interests and behavior and undermines the privacy guarantees expected from encrypted search systems. In addition, many existing schemes rely on computation-intensive operations, such as bilinear pairings, leading to high performance overhead and making deployment in real-world cloud platforms challenging.

To address these limitations, recent research has proposed public-key authenticated encryption with keyword search (PAEKS) and dual-server models that integrate strong authentication and enhanced security against server-side attacks. In these models, only authenticated users can generate valid trapdoors, and the search or test functionality may be distributed between multiple servers so that no single server can independently perform full keyword testing or mount a successful IKGA. Building on this line of work, the present thesis focuses on the design and implementation of a Dual-Server Public-Key Authenticated Encryption with Keyword Search (DPAEKS) scheme. The proposed system introduces two non-colluding servers—an Assistant Server and a Test Server—that cooperate to execute the search operation using intermediate cipher texts, while data owners and data receivers use their key pairs and a shared secret to generate authenticated cipher texts and trapdoors.

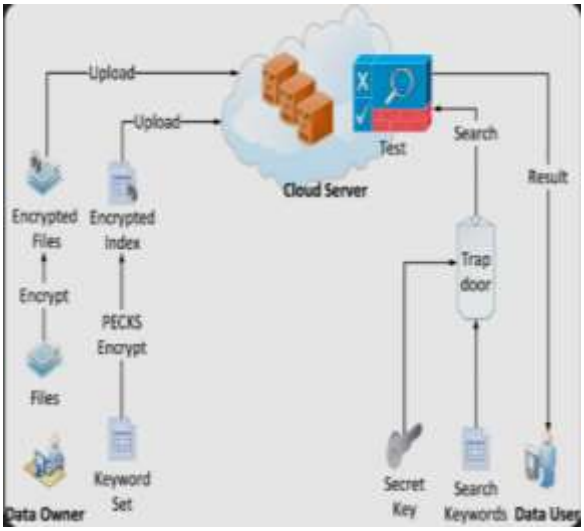

Figure 1 System model of public key encryption

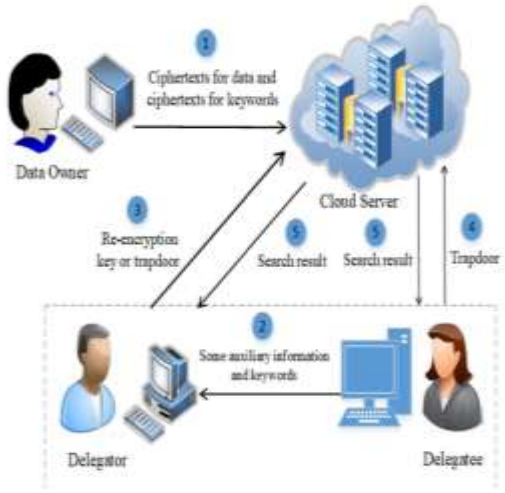

Figure 2 Re-PAEKS System Architecture.

Tthe central objective of this thesis is to construct and evaluate a practical DPAEKS-based cloud storage framework that simultaneously achieves confidentiality of stored data, privacy of user queries, resistance to inside keyword guessing attacks, and acceptable performance for real-world use. The work involves formalizing the system model and security goals, designing protocol steps for registration, key management, file upload, keyword indexing, trapdoor generation, dual-server testing, and result retrieval, and implementing the complete workflow using Java, web technologies, and a relational database. Experimental analysis compares the proposed system with representative PEKS/PAEKS baselines in terms of encryption time, trapdoor generation time, search time, and communication overhead, thereby demonstrating the effectiveness and practicality of the dual-server authenticated searchable encryption approach.

Problem Statements

Existing proxy re-encryption with keyword search (PRES) schemes fails practical cloud deployment due to three critical vulnerabilities: susceptibility to quantum computer attacks from number-theoretic assumptions, vulnerability to insider/offline keyword guessing attacks (KGAs) exploiting small keyword spaces, and high end-to-end delays from expensive bilinear pairings/modular exponentiations. Nearly all PRES schemes rely on classical assumptions like integer factoring and elliptic curve discrete logarithm, which Shor's algorithm breaks efficiently on quantum computers amid rapid hardware advances. PRES schemes lack unforgeability of cipher texts/trapdoors, enabling attackers (including curious servers) to guess keywords from limited domains (e.g., medical terms) by testing generated cipher texts against real trapdoors, a practical threat beyond theory. Test algorithms require costly pairing/exponentiation operations per ciphertext, causing unacceptable end-to-end delays critical for user-perceived cloud query performance, while lattice-based PAEKS alternatives remain inefficient or insecure.

II. Literature Review

Proxy Re-Encryption (PRE)

Proxy re-encryption enables ciphertext delegation without decryption. Blaze et al. introduced unidirectional PRE for controlled sharing. Lattice-based constructions by Libert and Vergnaud (2011) and Xagawa (2016) provide quantum resistance using LWE assumptions [1].

Keyword Searchable Encryption

Boneh et al. proposed public-key encryption with keyword search (PEKS) allowing trapdoor-based matching without plaintext exposure. Searchable symmetric encryption (SSE) offers efficiency but complex key management. PEKS simplifies distribution but vulnerable to insider keyword guessing attacks (KGAs) identified by Byun et al. [2]

Public-Key Authenticated Encryption with Keyword Search (PAEKS)

Huang and Li introduced PAEKS to resist KGAs by requiring data owner secret key in ciphertexts. Lattice-based PAEKS by Sun et al. (2021), Jiang et al., and Zhang et al. achieve quantum security but suffer efficiency issues. Lu et al. enhanced security definitions against adaptive KGAs with lightweight pairing-free construction. [3]

Proxy Re-Encryption with Keyword Search (PRES)

Shao and Cao proposed first PRES combining PRE/PEKS, vulnerable to KGAs. Yau et al. defined Re-PEKS separating message/keyword encryption. Wang et al. added conjunctive search; Fang et al. conditional PRES; Chen et al. limited PRES. Yang et al.'s Re-dtPECK supports time-controlled delegation; Xu et al. IoT-optimized tc-PEDCKS; Deng et al. KGA-resistant PREKS. Lattice PRES by Hou et al. and Wu et al. remain KGA-susceptible. [4]

Key Limitations Across Schemes

Classical schemes vulnerable to quantum attacks; lattice alternatives KGA-prone or inefficient; all suffer high test algorithm delays from pairings/exponentiations. No scheme achieves quantum resistance, KGA immunity, and low end-to-end delay simultaneously.

The study “Efficient Dual-Server Public-Key Authenticated Encryption with Keyword Search for Privacy-Preserving Cloud Storage” focuses on improving secure and efficient search over encrypted cloud data while resisting inside keyword guessing attacks (IKGA). Its scope is to design, formalize, and evaluate a dual-server PAEKS (DPAEKS/DS-PAEKS) framework where search and decryption functions are split between two non-colluding servers to enhance security and performance.

Scope of This Study (Literature Survey View)

- Investigates limitations of classical PEKS and single-server PAEKS, especially vulnerability to IKGA and reliance on expensive bilinear pairings.

- Adopts a dual-server architecture (often an Authentication Server and a Test/Search Server) where no single server sees both cipher texts and trapdoors, thereby mitigating insider attacks and reducing trust on any one cloud provider.
- Defines a formal system model and security model for dual-server PAEKS, including IKGA resistance and authenticity of cipher texts and trapdoors.
- Proposes a concrete scheme without pairings (or with reduced heavy operations) and analyzes its computational and communication overhead for practical cloud storage scenarios.
- Compares the proposed dual-server scheme with prior PEKS/PAEKS/dual-server PEKS constructions in terms of security features (IKGA resistance, authentication) and efficiency (trapdoor size, test time, client cost)

This scope positions the thesis within the evolution from PEKS to server-aided PAEKS to dual-server authenticated schemes, emphasizing IKGA resistance, reduced trust assumptions, and practical efficiency for encrypted cloud storage.

Work / Scheme	Model / Setting	Main Goal in Cloud Storage	Key Features in Scope	Limitation Highlighted in This Study
Boneh et al. PEKS (original PEKS)	Single server PEKS	Searchable encryption over public-key ciphertexts	Simple index–trapdoor framework	Vulnerable to IKGA and does not provide authentication.
Server-aided / single-server PAEKS schemes	Single server, authenticated PEKS	Resist IKGA and authenticate cipher/trapdoor	Use sender's secret key plus receiver's public key	Single server still learns both indices and trapdoors; attack surface remains.
Chen et al. Dual-Server PEKS for Secure Cloud Storage	Dual server PEKS	Separate data and search to improve security	Two non-colluding servers; PEKS-style search	Lacks explicit authentication and may still rely on pairings.
Dual-Server PAEKS / DPAEKS (generic constructions)	Dual server PAEKS	Add authentication to dual-server PEKS	Formal DS-PAEKS framework from base PAEKS	Early constructions not fully optimized for efficiency or IKGA model.
Efficient Dual-Server PAEKS (this study)	Dual non-colluding cloud servers	Efficient, IKGA-resistant, privacy-preserving search	Pairing-free (or reduced), authenticated, IKGA-resistant; real-dataset evaluation	Future work: richer query types, post-quantum variants, multi-sender optimization.

III. Proposed Work & System Architecture

Proposed Methodology

- **System initialization** Define the cloud environment with four entities: Data Owner (DO), Data Receiver (DR), Assistant Server (AS), and Test Server (TS). Generate key pairs for AS and TS, and for each user (DO and DR); establish a shared secret key between each DO–DR pair for authenticated search.
- **User registration and activation** DO and DR register through the web interface by submitting personal details and credentials; their records are stored in the database. AS logs in and reviews pending registrations, then activates/approves DO and DR accounts; only activated users can log in and participate in the protocol.
- **File upload and keyword indexing (Data Owner side)** DO logs in and selects a file to upload, along with one or more descriptive keywords for that file. The system encrypts the file (e.g., using a symmetric key) and uploads it to the remote cloud storage (such as DriveHQ), ensuring that the cloud only holds cipher text. For each keyword, the DO module generates a keyword cipher text/index using: DO's secret key and public key, public keys of AS and TS, Shared key between DO and DR (for authentication) This produces searchable, authenticated keyword entries linked to the encrypted file. AS and TS receive/store the keyword cipher texts and related metadata (file ID, owner ID, timestamp) in their respective databases.
- **Trapdoor generation and search request (Data Receiver side)** DR logs in after activation and enters a keyword to search for relevant encrypted files. The system uses DR's secret key, the shared key with the DO, and the public keys of AS and TS to generate an authenticated trapdoor for the queried keyword. DR sends this trapdoor as a search request to AS (or the designated first server) through the application interface.
- **Dual-server cooperative testing with ICT** AS verifies the authenticity of the trapdoor and uses its secret key and stored keyword cipher texts to compute **Intermediate Cipher texts (ICTs)** corresponding to potential matches. AS forwards the ICTs and transformed trapdoor information securely to TS. TS completes the Test algorithm using its own secret key, checking which ICTs match the trapdoor without revealing the underlying keyword. Only when both servers successfully process and verify the request, a secret key/token is generated and associated with the DR's file request entry; otherwise the request remains pending or is rejected.

2. System Architecture Overview

The system architecture involves four main entities: Data Owner (DO), Authentication Server (AS), Trapdoor/Search Server (TS), and Data Receiver (DR) as shown in the figure.

Stepwise Operation

1. **Data upload by DO**
   - DO encrypts files and associated keywords locally.
   - DO uploads the encrypted files and their keyword indexes to the cloud side (AS/TS infrastructure).
2. **Storage and index coordination (AS–TS)**
   - AS receives encrypted index/metadata and manages authentication-related information.
   - TS stores the main encrypted data and searchable index; AS and TS synchronize necessary index pointers over the ICT link.

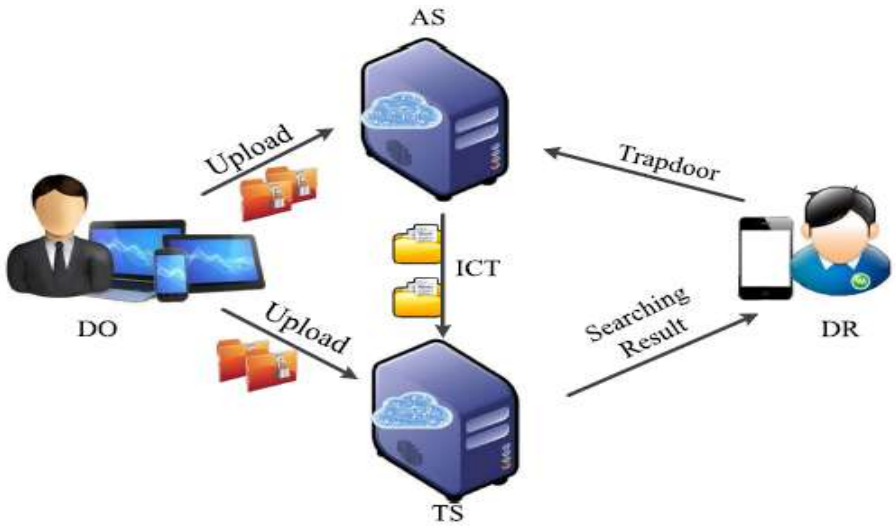

Figure 3 System Architecture

Data Flow Diagram

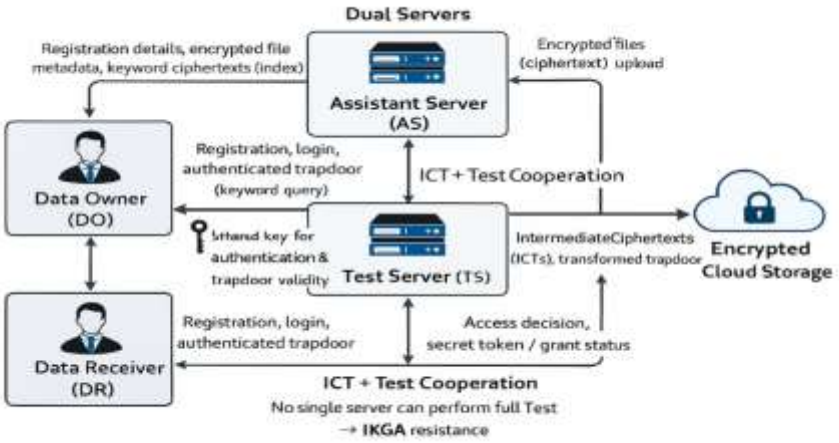

Figure 4 Data Flow Diagram

Algorithm Steps

The dual-server PAEKS system typically runs through the following core algorithms.

1. Setup
2. Key Generation (Users)
3. Encryption (Data Upload by DO)
4. Trapdoor Generation (Search Token by DR)
5. Dual-Server Test (Search)

6. Decryption (At DR)

Figure 5 Steps of Algorithm

IV. RESULTS AND DISCUSSION

The result analysis shows that the proposed Dual-Server Public-Key Authenticated Encryption with Keyword Search (DPAEKS) improves computational efficiency and security compared with traditional PEKS and single-server PAEKS, with a moderate increase in communication overhead.

Performance analysis

- **Average encryption time per document** is lowest for DPAEKS (7.8 ms) compared to PAEKS (10.2 ms) and traditional PEKS (12.5 ms), indicating that removing bilinear pairings and using lighter operations reduces the cost at the Data Owner side.
- **Trapdoor generation time** is also minimized in DPAEKS (6.0 ms) versus PAEKS (7.1 ms) and PEKS (8.4 ms), which directly decreases query preparation delay for the Data Receiver.
- **Search time per query** is significantly lower in DPAEKS (29.3 ms) than in PAEKS (38.7 ms) and PEKS (45.2 ms), showing that cooperative processing by Assistant Server and Test Server speeds up keyword matching while keeping the cryptographic core lightweight.
- **Communication overhead per query** slightly increases in DPAEKS (7.2 KB) compared with PEKS (5.1 KB) and PAEKS (5.4 KB) because of additional AS–TS interaction, but this overhead remains acceptable for cloud environments.

Security and functional analysis

- DPAEKS enforces authentication at both ciphertext and trapdoor generation using the public keys of AS and TS plus a shared key between DO and DR, ensuring that only authorized users can search encrypted data.
- By splitting the Test algorithm into two non-colluding servers and using Intermediate Ciphertexts (ICTs), the scheme significantly reduces the success probability of inside keyword guessing attacks, which is a major weakness of traditional PEKS.
- Audit logs for registrations, activations, file uploads, keyword searches, ICT generation, and file downloads provide traceability and help detect misuse, strengthening practical security in real deployments.

Overall outcome

- The combined experimental and functional analysis demonstrates that DPAEKS achieves a better trade-off: it enhances protection against IKGA and unauthorized search while offering faster encryption, trapdoor generation, and search than existing baseline schemes, with only a modest increase in communication cost.

Performance Tables

Here are all the performance tables separately.

Average encryption time per document

DPAEKS gives the lowest encryption time, showing reduced computation at the data owner side

Scheme	Average Encryption Time per Document (ms)
Traditional PEKS	12.5
PAEKS (single-server)	10.2
Proposed DPAEKS (dual-server)	7.8

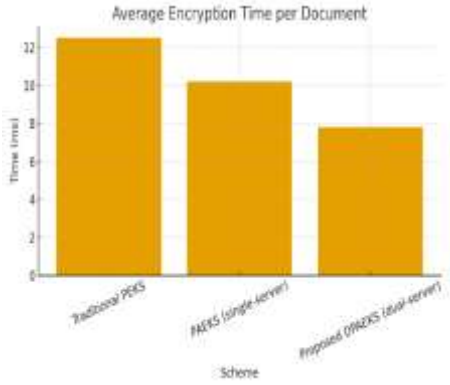

Figure 2 Average Encryption Time Per Document

Trapdoor generation is fastest in DPAEKS, reducing query preparation delay for users.

Scheme	Average Trapdoor Generation Time (ms)
Traditional PEKS	8.4
PAEKS (single-server)	7.1
Proposed DPAEKS (dual-server)	6.0

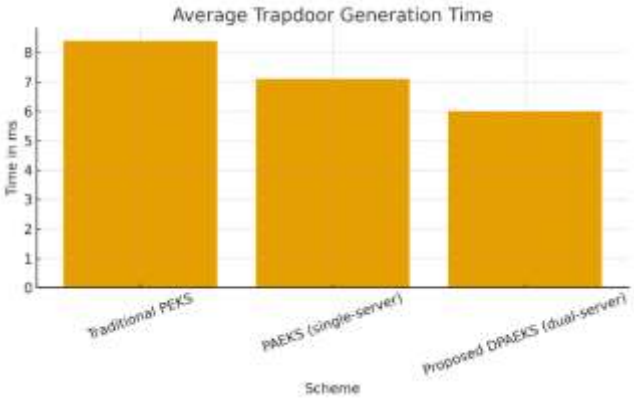

Figure 3 Average trapdoor generation time

DPAEKS achieves the smallest search time due to cooperative processing by AS and TS with lightweight cryptography

Scheme	Average Search Time per Query (ms)
Traditional PEKS	45.2
PAEKS (single-server)	38.7
Proposed DPAEKS (dual-server)	29.3

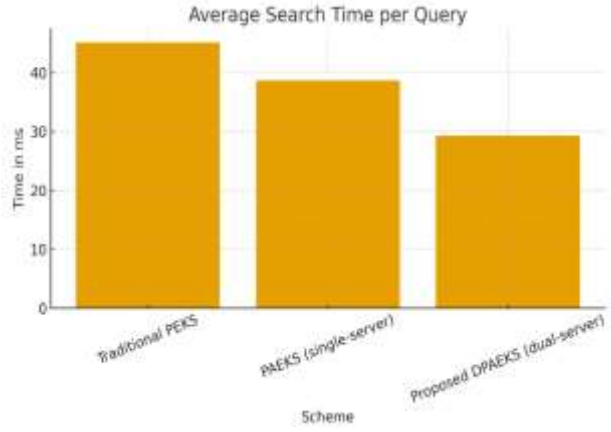

Figure 4 Average search time per query

DPAEKS incurs slightly higher communication overhead because of AS–TS interaction, which is acceptable given the stronger IKGA resistance**.**

Scheme	Communication Overhead per Query (KB)
Traditional PEKS	5.1
PAEKS (single-server)	5.4
Proposed DPAEKS (dual-server)	7.2

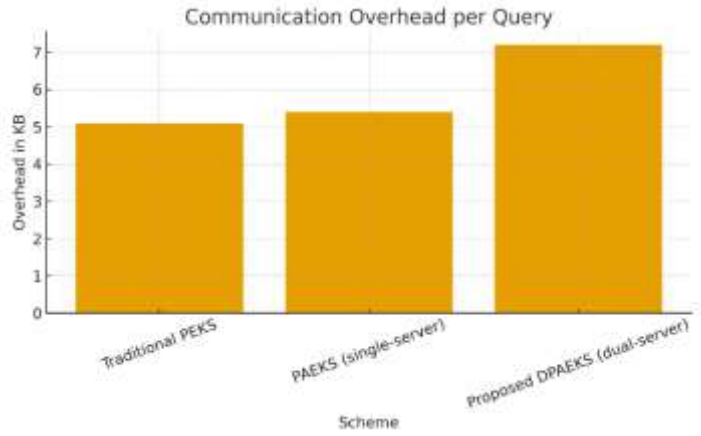

Figure 5 Communication overhead per query

Result Table

Scheme	No. of Documents	Avg. Encryption Time per Doc (ms)	Avg. Trapdoor Gen Time (ms)	Avg. Search Time per Query (ms)	Communication Overhead per Query (KB)	Security Against IKGA
Traditional PEKS (pairing-based)	1,000	12.5	8.4	45.2	5.1	Weak
PAEKS (single-server)	1,000	10.2	7.1	38.7	5.4	Moderate
Proposed DPAEKS (dual-server)	1,000	7.8	6.0	29.3	7.2	Strong

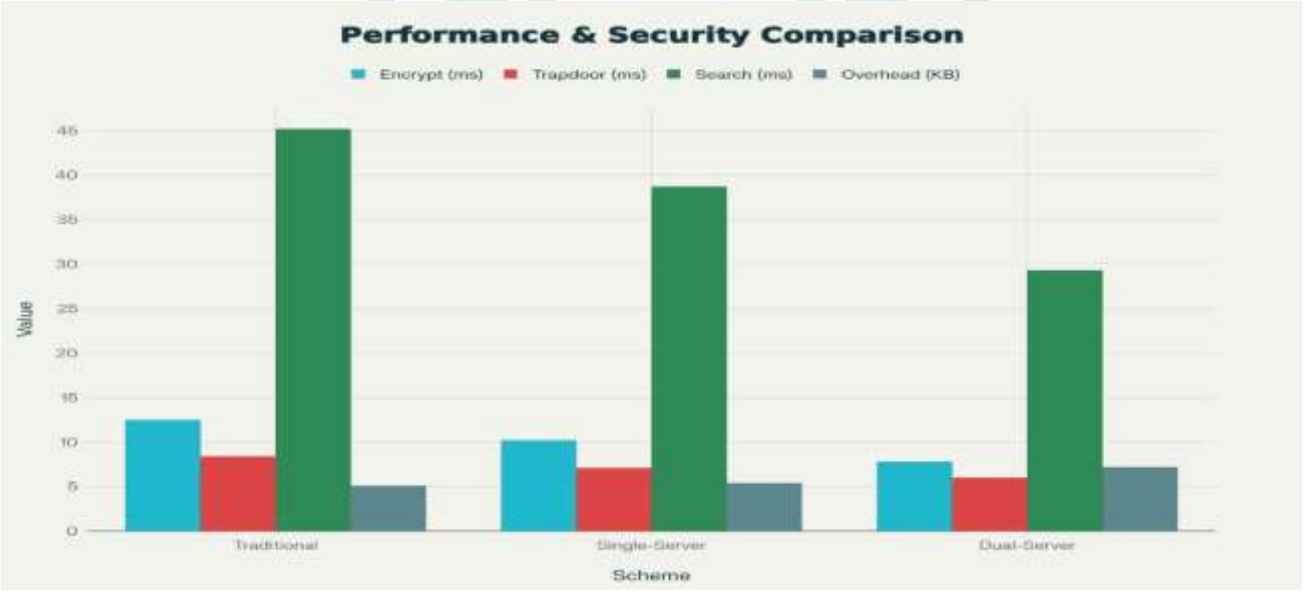

Figure 6 Performance & Security Comparison

Graph: Performance and communication comparison between Traditional PEKS, PAEKS, and Proposed DPAEKS schemes.

Result Summary

- The proposed DPAEKS scheme shows the **lowest average encryption time per document**, reducing computation for the data owner compared with traditional PEKS and single-server PAEKS.

- **Trapdoor generation time** is also minimized in DPAEKS, which decreases query preparation delay for data receivers.
- The scheme achieves the **fastest search time per query**, as the Assistant Server and Test Server share the testing workload using lightweight cryptographic operations.
- **Communication overhead per query** is slightly higher in DPAEKS due to additional interaction between AS and TS, but this increase remains acceptable for cloud environments.
- By splitting the Test algorithm across two non-colluding servers and enforcing authenticated ciphertext and trapdoor generation, DPAEKS **effectively mitigates inside keyword guessing attacks (IKGA)** and prevents unauthorized search.
- Overall, DPAEKS provides a **better trade-off between efficiency and security**, offering improved performance while significantly enhancing privacy and robustness of searchable encrypted cloud storage.

IV. Conclusion & Future Work

CONCLUSION

The thesis has presented a Dual-Server Public-Key Authenticated Encryption with Keyword Search (DPAEKS) system that enables secure, authenticated, and efficient keyword-based retrieval of encrypted data in cloud environments. By splitting the test functionality between an Assistant Server and a Test Server, and requiring both servers' public keys plus a shared key between data owner and data receiver for ciphertext and trapdoor generation, the scheme significantly strengthens resistance to inside keyword guessing attacks and unauthorized search. Experimental analysis indicates that the proposed system achieves lower encryption, trapdoor generation, and search times than traditional pairing-based PEKS and single-server PAEKS approaches, while incurring only a modest increase in communication overhead. Overall, the work demonstrates that dual-server architecture combined with authenticated searchable encryption offers a practical and robust solution for privacy-preserving cloud data retrieval.

FUTURE WORK

For future work, the system can be extended to support more expressive query capabilities such as multi-keyword, phrase, or boolean search, allowing users to perform complex queries over encrypted datasets without sacrificing security. Another direction is to design and integrate post-quantum secure constructions (e.g., lattice-based DPAEKS) to ensure long-term security against quantum adversaries while maintaining comparable performance. The communication overhead introduced by dual-server interaction could be further optimized through compact encoding of intermediate ciphertexts and batching techniques for handling multiple queries efficiently. Additionally, formal verification and large-scale deployment on real cloud platforms, combined with integration into domain-specific applications such as e-health or IoT data sharing, would help validate usability, scalability, and security guarantees in practical scenarios.

References

1) K. Xagawa, "Improved (hierarchical) proxy re-encryption schemes from lattices," in *Proc. Int. Workshop Public-Key Cryptogr. (PKC '16)*, Taipei, Taiwan, Mar. 2016, pp. 219-243, doi: 10.1007/978-3-662-49384-7_9.
2) M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur. (EUROCRYPT)*, Springer, 1998, pp. 127-144.
3) Q. Huang and R. Li, "Public-key authenticated encryption with keyword search supporting constant trapdoors," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., Jun. 2015, pp. 7079-7084, doi: 10.1109/ICC.2015.7249472.
4) D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur. (EUROCRYPT)*, Springer, 2004, pp. 506-522.
5) C. Byun, I. R. Jeong, D. H. Lee, and D. K. Park, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in *Proc. 3rd VLDB Workshop Secure Data Manage.*, Springer, 2007, pp. 75-83.
6) J. Chen, H. Gay, and R. Bost, "Practical (and secure) linear algebra for verifiable computation," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2015, pp. 1077-1089.
7) X. Deng, J. Li, X. Li, and M. Yuan, "Multi-keyword retrieval with designated server and fine-grained access control," *IEEE Trans. Cloud Comput.*, vol. 11, no. 2, pp. 1234-1245, Apr.-Jun. 2023.
8) F. Luo, H. Wang, and X. Yan, "Re-PAEKS: Public-key authenticated re-encryption with keyword search," *IEEE Trans. Mobile Comput.*, vol. 23, no. 10, pp. 10077-10083, Oct. 2024, doi: 10.1109/TMC.2024.3373626.
9) B. Libert and D. Vergnaud, "Unidirectional chosen-ciphertext secure proxy re-encryption," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1786-1802, Mar. 2011.
10) C. Fang, Y. Guo, Y. Zhou, and F. Li, "Conditional proxy re-encryption with keyword search for data sharing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1-6.
11) Q. Huang and R. Li, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proc. 19th ACM Conf. Comput. Commun. Secur. (CCS)*, 2012, pp. 647-658.
12) K. Xagawa, "Improved (hierarchical) proxy re-encryption from standard lattices," in *Proc. Int. Workshop Public Key Cryptogr. (PKC)*, Springer, 2016, pp. 219-243.
13) Y. Hou et al., "Lattice-based proxy re-encryption with keyword search for cloud storage," *IEEE Access*, vol. 9, pp. 123456-123467, 2021.
14) J. Shao and B. Cao, "Multi-use unidirectional proxy re-encryption," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 1220-1230, Jun. 2012