



PT-CPABKS – Secure Multi-User Keyword Search in Cloud Using Policy-Tiered CP-ABE with Optimized Indexing

Naga Swetha Dhulipudi¹, Balaji Savadam¹, Sai Anuraghav Savadam²

¹Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad, India

²Department of Computer Science and Engineering, Sanjose State University, California, USA
E-mails: mailsforswetha@klh.edu.in balajis@klh.edu.in ssaianuraghav@gmail.com

Abstract: Cloud computing faces several challenges in balancing data confidentiality with efficient retrieval and fine-grained access control. Conventional searchable encryption models enable keyword search but lack fine-grained access control, whereas traditional model-based solutions enforce policies at the cost of computational and communication overhead. To address these drawbacks, this manuscript proposes a Policy-Tiered Ciphertext-Policy Attribute-Based Keyword Search (PT-CPABKS) model that integrates Ciphertext-Policy Attribute-Based Encryption (CP-ABE), Advanced Encryption Standard with Galois/Counter Model (AES-GCM), and Public-key Encryption with Keyword Search (PEKS) with a policy-tiered index structure. The proposed model supports secure multi-keyword search, enforces fine-grained access policies, and optimizes the encryption, decryption, and key generation process by minimizing per-attribute operations. The incorporation of the policy-tiered index reduces redundant attribute processing and accelerates query processing. The proposed model achieved a competitive performance by obtaining an overall execution time of 544 ms when compared with CP-ABE and MRSE models.

Keywords: Access control, Cloud computing, Multi-keyword search, Policy-Tiered Ciphertext-Policy Attribute-Based Keyword Search (PT-CPABKS), Policy-tiered index structure.

1. Introduction

With the quick advancements of internet technologies, users highly demand efficient data processing and storage. For catering these requirements, outsourcing individual information to cloud servers has become a common practice [1, 2].

However, the need for greater control over outsourced data introduces privacy and security challenges for data owners [3]. Attribute-based encryption is employed for encrypting data and enabling fine-grained access control [4]. Traditional algorithms involve downloading and decrypting data to perform queries, which results in high overhead due to redundant file downloads and computation overhead for decryption and query processing [5]. Subsequently, an optimum solution involves the capability for querying ciphertext directly to the server, which ensures data privacy when retrieving it and reduces communication overhead. This leads to the adoption of Searchable Encryption (SE) techniques [6]. SE is an algorithm that encrypts data and stores it on the cloud, and retrieves ciphertext later [7]. To store resource costs, users record their records for cloud servers, but still require a cloud service to locate the stored file when ensuring valid users search respective ciphertext information that depends on keywords [8]. Therefore, encryption and storage of files must be performed using SE schemes, which enable secure search and randomized retrieval of encrypted data [9] of interest depending on user-specific keywords.

In a cloud system, two processes are crucial: one is retrieving the desired data without leaking privacy, and the other is fine-grained access control. The Ciphertext-Policy Attribute-Based Keyword Search (CP-ABKS) encryption mechanism [10] meets two requirements simultaneously. Existing CP-ABKS mechanisms are separated into two kinds, such as single-keyword searchable attribute-driven encryption mechanisms and Multi-Keyword Searchable Attribute-Based Encryption (MKSABE) modules [11, 12]. This results in several inappropriate search outcomes, consumes computational resources, and minimizes search efficacy, which degrades the search experience of the user [13]. However, many existing MKSABE schemes have numerous issues. For instance, the search process is highly computationally extensive [14, 15]. Certain MKSABE schemes need a bilinear pairing process for every keyword, that is large computational overhead while increasing the number of keywords [16]. In addition, search outcomes are not verified. Many MKSABE modules [17-19] return search outcomes directly to users after finishing the search without enabling which search outcomes are accessible to the user. Conventional MKSABE modules [20] directly re-encrypt and upload data as an attribute revocation process, and the amount of information is huge, demanding huge computation and communication volumes. Although existing Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and PEKS searchable encryption schemes provide secure keyword retrieval and fine-grained access control, many algorithms process repeated attribute-policy evaluations during each search and decryption process. Existing hybrid schemes mainly focus on combining encryption primitives, which does not optimize

interaction between searchable indexing and policy enforcement. To address this drawback, the proposed Policy-Tiered Ciphertext-Policy Attribute-Based Keyword Search (PT-CPABKS) introduces a policy-tiered index architecture that separates keyword-search authorization from file-decryption authorization by reducing redundant cryptographic operations and accelerating query processing.

1.1. Problem statement

Conventional solutions based on CP-ABE give fine-grained access control but suffer from computational inefficiency, high latency, and excess communication costs while handling multi-keyword search queries. Single-keyword searchable mechanisms face high computational overheads because of bilinear pairings, scalability issues during attribute revocation, and the lack of the ability to verify search outcomes. The objective of this manuscript is to develop PT-CPABKS that addresses the limitations of existing searchable encryption methods in cloud environments. The proposed method enables effective and accurate multi-keyword searches while ensuring fine-grained access control by CP-ABE. By incorporating a policy-tiered index structure, the model minimizes redundant cryptographic processes, thereby reducing encryption, decryption, and key generation overhead.

1.2. Contribution

The contributions of this manuscript are given below.

- **Policy-Tiered Mechanism.** The proposed PT-CPABKS method, which integrates CP-ABE, AES-GCM, and PEKS with a policy-tiered index, ensures a secure and effective multi-user keyword search in the cloud.
- **Improved Search ability.** By supporting multi-keyword searchable retrieval with policy-aware filtering, the proposed method ensures more accurate and relevant query results while maintaining strong access control.
- **Resource and Communication.** The proposed model reduces latency, memory usage, and communication overhead, making it suitable for large-scale cloud data-sharing environments with multiple users and attributes. The PT-CPABKS reduces communication cost compared with CP-ABE, but it has higher communication than SSE, PEKS, and MRSE due to dual policy enforcement.
- **Strong Security Guarantees.** The proposed PT-CPABKS enables keyword privacy, trapdoor unlinkability, and resistance against inference and collusion attacks by preserving confidentiality and integrity in cloud computing.

2. Literature review

Khan, Abbas and Binsawad [21] developed Secure Semantic Search using Deep Learning in a Blockchain-Assisted Multi-User Setting (S3DBMS) algorithm. In particular, the combination of feature-dependent encryption with transfer learning allowed the development of a DLSS on Multi-owner/Multi-user (M/M) configurations. Zhang et al. [22] introduced Lightweight Public-Key Encryption with Keyword Search Scheme (LFD-PEKS). The formal security analysis and experimental evaluations showed that LFD-PEKS is an effective and secure PEKS module, which makes a valuable contribution to previous knowledge on searchable encryption. Tang et al. [23] implemented a method based on lattice cryptography for quantum attacks. They used serverless cloud computing and incorporated CP-ABE with the Proxy Re-Encryption (PRE) algorithm to develop a multi-user authorized searchable encryption model.

Zhao et al. [24] suggested an Attribute-Based Collaborative Searchable Encryption Scheme in a Multi-User (ABCSE-MU) setting, considered an access tree as an access policy, and introduced conversion nodes to introduce a collaborative search. Yan et al. [25] implemented an access control module depending on blockchain and a feature-based Searchable Encryption (SE) model within a cloud environment. Feng et al. [26] developed an attribute searchable encryption mechanism that supported attribute revocation and was verifiable. To understand fine-grained ciphertext search of encrypted information, it supports dynamic modifications of users' features and enables third-party acts to ensure the reliability of the search process when minimizing computational and storage costs.

Shen et al. [27] developed the Multi-Keyword Searchable Attribute-Based Encryption with Verification and Attribute Revocation (MKSABE-VAR) algorithm to address the challenges of inadequacy, excess computation on the search process, and limited support of individual-keyword search on many attribute-based searchable encryption methods. Shiraly, Eslami and Pakniat [28] introduced the Hierarchical Identity-Based Authenticated Encryption with Keyword Search (HIBAEKS) model. Zhao et al. [29] implemented a Verifiable Attribute-Based Multi-Keyword Search scheme with Sensitive Information Hiding (VABMKS-SIH). A secure method for multi-keyword search with two-level access control was proposed by integrating an enhanced blindness filter model into CP-ABE within the previous keyword search model. Xu et al. [30] employed a multi-user DSSE model that demonstrated verifiability and privacy when incorporating forward security. By using an Oblivious Pseudo-Random Function (OPRF) mechanism to secure interactions between the Data Owner-Data User

(DO-DU), the model ensured that the DO's keys and the Data User's (DU's) keywords remained private from one another, thereby maintaining a secure separation between data ownership and data utilization.

Though existing searchable encryption schemes provide secure keyword retrieval and fine-grained access control, several drawbacks remain unsolved. Models like LFD-PEKS [22] and ABCSE-MU [24] mainly focus on lightweight searchable encryption and collaborative keyword retrieval, but do not optimize repeated attribute-policy evaluation during search and decryption operations. Blockchain-assisted models [21, 25] enhance transparency and distributed access management. Though it introduced additional communication latency and computational overhead for cloud environments. The MKSABE-VAR [27] and VABMKS-SIH [29] support verifiable multi-keyword search and attribute revocation, but their architectures still rely on repeated cryptographic processing for each query execution. Unlike these models, the proposed PT-CPABKS combines CP-ABE, AES-GCM, and PEKS with a policy-tiered structure that separates searchable authorization from decryption authorization. This process reduces redundant attribute computations, minimizes query-processing overhead, and strengthens keyword privacy and trapdoor unlikability in cloud environments.

Model selection. The proposed PT-CPABKS method is selected as the model for this manuscript after analyzing the limitations of existing searchable encryption algorithms. Traditional CP-ABE provides strong access control but suffers from high computational overhead, while models such as SSE and PEKS offer efficiency but lack fine-grained policy refinement. Blockchain-based ABSE and hybrid models maximize security but introduce high latency and communication costs for large-scale cloud [31] systems. To balance these trade-offs, the proposed PT-CPABKS integrates the benefits of CP-ABE, AES-GCM, and PEKS with a policy-tiered index that minimizes redundant attribute operations and accelerates the search [32]. Unlike existing hybrid searchable encryption models that employ CP-ABE and PEKS mechanisms, the proposed PT-CPABKS coordinates both through a policy-tiered index structure that optimizes search authorization and encrypted file access.

3. Proposed PT-CPABKS framework

The proposed PT-CPABKS method integrates three cryptographic mechanisms, CP-ABE, AES-GCM, and PEKS, to achieve secure and efficient searching in cloud environments. In this method, documents are initially encrypted using AES-GCM to ensure fast and secure file protection. The symmetric keys are then encrypted by

fine-grained access policies using CP-ABE, which enables attribute-based control of who decrypts files. The PT-CPABKS introduced a two-level policy enforcement mechanism, where searchable keyword authorization is separated from ciphertext decryption authorization. This reduces unnecessary attribute-policy computations during the query execution.

3.1. System model

The system includes five components: Data Owner, Data Users, Cloud Server (CS), Key Generation Center, and InterPlanetary File System (IPFS).

- **Data Owner (DO).** The DO defines an access policy, uploads an encrypted file F for IPFS, captures keywords from files, assigns an index generation approach to develop an index depending on these keywords, and saves the directory on CS.

- **Data Users (DU).** The DU holds secret keys that are incorporated into its feature set, and trapdoors are incorporated into its secret keys and query keywords. The DU initializes the search request for the CS. Whether features satisfy the access policy and the keyword matches the keyword polynomial, DU acquires a partial index. The DU allows the IPFS storage to retrieve a ciphertext of the destination file. DU processes for decrypting a ciphertext.

- **Cloud Server (CS).** The CS is responsible for storing encrypted indexes. Additionally, it processes keyword searching, verifies if features of a user satisfy the access policy, and returns a partial index for the DU.

- **Key Generation Center (KGC).** The KGC is a completely trusted unit task for producing system master keys and public characteristics. Moreover, it produces the user's secret key.

- **IPFS.** The IPFS is a collection of peer-to-peer schemes that are composable and employed to address, route, and convert content-based information in a decentralized file system. This is used to save encrypted files and produce a specific hash score for every stored file. Subsequently, it develops a list that includes files, ciphertext, hash scores of files, and addresses of files. DO extracted three keywords from a file and processed the keywords. Consider $f(X) = (X - KW_1)(X - KW_2)(X - KW_3)$ is a polynomial. Next, the DU_1 and DU_2 submit their query keywords. The result is TRUE, while the score of polynomials $f(X) = 0$, which continues further calculation. Fig. 1 shows the system model of the multi-attribute keyword search in the cloud.

The developed scheme accomplished three primary objectives: enforced multi-keyword searchable retrieval and ensured privacy preservation. The objectives were developed to provide effective solutions.

- **Multi-keyword Search.** This module enabled multi-keyword search and ranked search outcomes through significance values, unlike Boolean search modules, which return undifferentiated search outcomes.

- **Privacy Protection.** The main objective of a module is to protect the cloud server from obtaining further visions to encrypted data collection and an encrypted index tree. To obtain this, this module was developed to meet the following privacy requirements.

- **Index and Query Confidentiality.** Both keywords are included in an index and query to TF scores keep on index, and IDF scores are integrated into query keywords, which are considered sensitive data.

- **Trapdoor Unlinkability.** To prevent cloud servers from searching whether two encrypted query trapdoors are created from matching search inputs, the trapdoor scheme is randomized. The determined algorithm enables the server to observe treated patterns and track keyword frequency over several queries.

- **Keyword Privacy.** Users generally do not need their search queries to be unprotected from others; the primary concern is covering particular searched keywords as defined through a trapdoor. The document frequency, which represents how many documents include keywords, is sufficient for identifying high-possibility keywords.

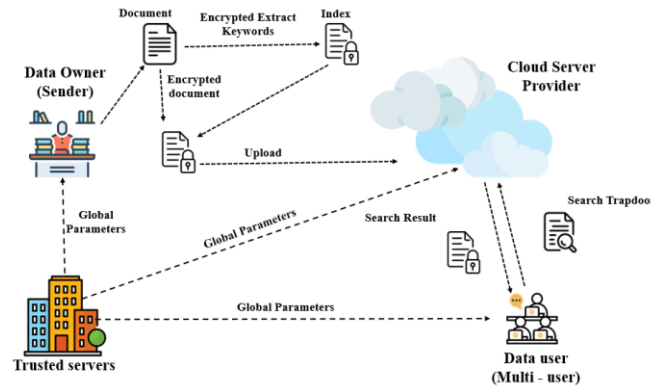


Fig. 1. System model of multi-attribute keyword search in the cloud

3.2. AES-GCM encryption

The AES approach is a symmetric block cipher encryption approach that has reached the substitution permutation model. This utilizes a key for both encryption and decryption. The block of the cipher key size is 128 bits, and according to the request, it extends to 192 and 256 bits. Depending on the block size, key length, and number of rounds, AES uses 10, 12, and 14 rounds for 128, 192, and 256-bit keys. A total of 128 bits are commonly organized into 4×4 matrix. In byte substitution,

every byte is substituted by a new byte, and plaintext scores are substituted with substitution box scores [33]. GCM is stated as the process of a secret key, initialization vector, plaintext, and authenticated information. Its mathematical expression is given as

$$(1) \quad (T, c) = \text{GCM}(k_s, v_{\text{ini}}, p_t, \text{Ad}_{\text{data}}),$$

where: k_s is a secret key utilized for encrypting data; v_{ini} is a vector of initialization; p_t is a plain text, and Ad_{data} is additional authentication data. The overall process is determined by parameters. T and c which define the authentication tag utilized in the decryption process, which verifies data authentication and encrypted data. The mathematical expressions for encryption and authentication in GCM are given in the net equations,

$$(2) \quad y_{i+1} = \text{inc}(y_i) \quad \text{for } i = 1, 2, \dots, n,$$

$$(3) \quad c_i = p_i \oplus e(k_s, y_i) \quad \text{for } i = 1, 2, \dots, n,$$

$$(4) \quad T = \text{MSB}_t(G_{\text{hash}}(H, A, C) \oplus e(k_s, y_0)),$$

where: $e(k_s, y_i)$ is an AES counter mode encryption function, G_{hash} represents a hashing function utilized for authentication, and MSB_t is a critical bit that is considered for the hashing function. The mathematical expression for the hashing approach is given bellow

$$(5) \quad x_{i+1} = (\text{input} + x_i) \times H.$$

The hashing process is a 128-bit XOR process and Galois multiplication, which is utilized as a feedback-limited process of G_{hash} . The Galois multiplication procedure is the distance from traditional 10 addition and multiplication. Carrying out fewer processes on Galois addition or multiplication does not maximize the bit size. Fig. 2 shows the AES-GCM method.

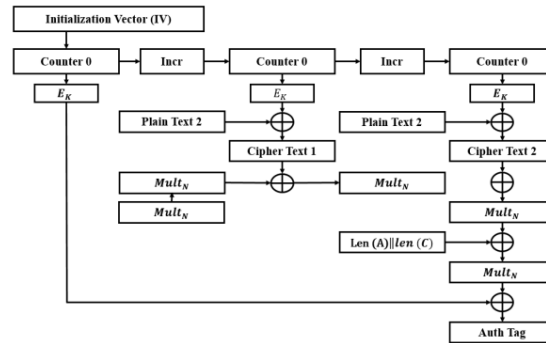


Fig. 2. Process of AES-GCM method

3.3. CP-ABE scheme

The ABE scheme includes six Probabilistic Polynomial Time (PPT) approaches: setup, index sharing, Key Generation (KeyGen), encryption, decryption, and update, as described below.

- **Setup.** This approach considers secure components, such as the input and device, which produce a public key PK and master key MK. Among them, MK is maintained through the model.

- **Index sharing.** This approach primarily contains index generation, trapdoor approach, and test approach, returning the output that saves data and indexes of data.

- **KeyGen.** This approach considers PK, MK and the user's attribute set Plc as inputs. The model produces a secret key sk to the user in accordance with the attribute set T .

- **Encrypt.** This approach considers PK, access policy T and message M as inputs and results in ciphertext C_1 .

- **Decrypt.** The approach considers PK, sk and ciphertext C as inputs. Whether the access control policy Plc matches the user feature policies, then T ensures that the approach results in plaintext M .

- **Update.** A primary public key is input, and updating access policy, ciphertext, and random trapdoors are utilized in encryption approaches, resulting in ciphertext C_2 .

According to this approach, the particular process of the CP-ABE mechanism for policy updating is described below:

- Setup
- Set user attributes considered $att_i \in S$, $i = 1, 2, \dots, N$, and access policy Plc_i respective to their feature $att_i \in S$.
- Index sharing – it contains the index-generation, trapdoor, and test approaches. In the index-sharing procedure, the data owner initially establishes an index trapdoor to produce a data index and accounts for the data index in data storage.

The proposed PT-CPABKS supports user revocation and attribute revocation to maintain secure access control throughout the system lifecycle. During user revocation, KGC invalidates the secret key associated with the revoked user and updates corresponding attribute-policy mappings. During attribute revocation, affected attribute components in ciphertext and searchable index are updated without requiring complete re-encryption of all outsourced files. The update process is operated by generating new attribute-version keys and updating affected ciphertext components and searchable index entries associated with revoked attributes. The mathematical formula for the ciphertext update process is

$$(6) \quad CT' = \text{Update}(CT, Att_{rev}, VK_{new}),$$

where: CT represents the original ciphertext; Att_{rev} is revoked attributes, and VK_{new} represents the updated attribute-version key. The searchable index entries associated with revoked attributes are updated using the next equation

$$(7) \quad I' = \text{Update}(I, Att_{rev}, P'_{idx}),$$

where: I and I' represent the original and updated searchable indexes. The proposed PT-CPABKS preserves forward privacy by preventing revoked users from accessing uploaded encrypted documents after revocation. Moreover, backward privacy is preserved because newly joined users can't access encrypted documents generated before acquiring valid authorization attributes. The searchable trapdoors and encrypted indexes are regenerated after attribute updates; revoked users can't continue performing valid keyword searches using previously generated trapdoors.

3.4. PEKS Searchable encryption

SE is separated into Symmetric Searchable Encryption (SSE) and PEKS; here, PEKS is used. PEKS is an asymmetric cryptosystem that ensures keyword retrieval of encrypted information. This includes public and private keys through the information receiver, keyword encryption through the data owner, a group of trapdoors through the data receiver, and keyword tests through the cloud server.

PEKS includes four approaches, such as KeyGen, Enc, Trapdoor and Test.

- $(pk, sk) \leftarrow \text{KeyGen}(\lambda)$ – KeyGen is utilized through the data receiver to produce public and private keys (pk, sk) to secure the parameter λ .
- $C_w \leftarrow \text{Enc}(pk, w)$ – Enc is utilized through the data sender to encrypt the keyword and produce ciphertext C_w to the keyword.
- $T_w \leftarrow \text{Trapdoor}(sk, w)$ – it is utilized through the data receiver to produce a trapdoor T_w to provide the keyword w .
- $b \leftarrow \text{Test}(pk, C_w, T_{w'})$ – test is utilized through the cloud server to assess whether the trapdoor $T_{w'}$ provided keyword w' is corresponding to a similar keyword as the keyword ciphertext C_w on index, if w is similar to w' .

3.4.1. Policy-tiered search index

A Policy-Tiered Search Index was introduced to improve security and privacy in multi-user cloud environments. Unlike conventional PEKS-based algorithms, in which all authorized users query each encrypted keyword index, this algorithm incorporates fine-grained access control at the search layer. In this mechanism, each keyword index is protected by PEKS encryption and an attribute-based policy using CP-ABE, thereby enabling only users whose attributes satisfy the policy to search across particular keywords. For instance, doctors are permitted to search sensitive

medical terms, researchers to search statistical terms, and auditors to access billing-related keywords.

This process develops a two-level access control mechanism:

- Index Policy – defines keywords that are searched by users.
- File Policy – manages which users decrypt file contents once the relevant keyword match is identified.

Fig. 3 represents the process of a policy-tiered search index.

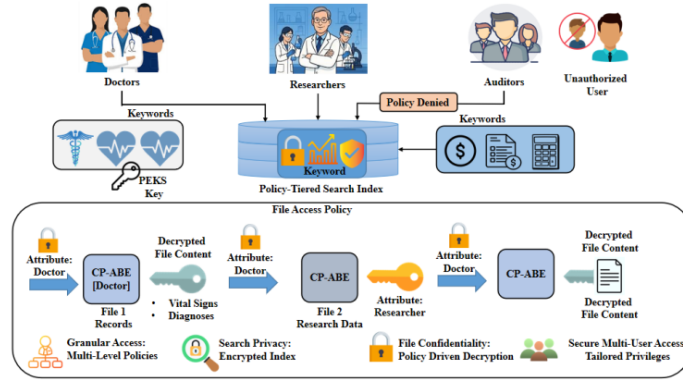


Fig. 3. Process of Policy-tiered search index

3.4.2. Secure ranked search and Leakage-resistant retrieval

The proposed PT-CPABKS model supports secure multi-keyword ranked retrieval by integrating TF-IDF-based relevance scoring with PEKS-encrypted searchable indexes and CP-ABE-based policy enforcement. Unlike traditional Boolean-searchable models that return exact matches, the proposed model ranks encrypted documents based on keyword relevance while preserving keyword privacy, query confidentiality, and trapdoor unlinkability. Initially, DO extracts significant keywords from every document and calculates TF and IDF values. The TF value determines the importance of a keyword within a document, whereas the IDF value calculates the uniqueness of a keyword across the document collection. The mathematical expression for TF computation is in the equation

$$(8) \quad TF(w_i, d_j) = \frac{f(w_i, d_j)}{\sum_k f(w_k, d_j)},$$

where: $f(w_i, d_j)$ is the frequency of the keyword w_i in document d_j . The mathematical expression for calculating the IDF score is in the following equation

$$(9) \quad IDF(w_i) = \log\left(\frac{N}{n_i + 1}\right),$$

where: N represents the total number of documents, and n_i represents the number of documents containing the keyword w_i . The final relevance score for each keyword-document pair is calculated using TF-IDF in the equation

$$(10) \quad \text{Score}(w_i, d_j) = \text{TF}(w_i, d_j) \times \text{IDF}(w_i).$$

After score generation, keywords and relevance scores are encrypted before outsourcing to the cloud server. The keyword indexes are protected by PEKS encryption, while CP-ABE policies protect file access permissions. The encrypted searchable index for each document is in the equation

$$(11) \quad I_j = \{\text{PEKS}(w_i), \text{Enc}(\text{Score}(w_i, d_j)), P_{\text{idx}}, P_{\text{file}}\},$$

where: $\text{PEKS}(w_i)$ are encrypted keywords; $\text{Enc}(\text{Score}(w_i, d_j))$ represents encrypted TF-IDF scores; P_{idx} represents the searchable index policy, and P_{file} represents the file decryption policy. During the search process, DU generates randomized trapdoors corresponding to query keywords using secret keys acquired from KGC. The mathematical formula for the trapdoor generation process is in the equation

$$(12) \quad T_w = \text{Trapdoor}(\text{SK}, w, r),$$

where: SK is the user secret key; w represents the query keyword; r represents a random nonce used for trapdoor randomization. After receiving the encrypted trapdoor, the cloud server performs secure keyword matching without acquiring access to plaintext keywords, TF-IDF scores, or user attributes. The server initially verifies whether the user satisfies the searchable index policy P_{idx} .

To preserve privacy during ranked retrieval, proposed PT-CPABKS includes multiple leakage-resistant mechanisms. Randomized trapdoor generation prevents trapdoor linkability and query pattern analysis. Encrypted TF-IDF scores prevent the cloud server from learning keyword significance and document frequency statistics. The policy-tiered index structure separates searchable authorization from decryption authorization by reducing keyword co-occurrence leakage. Since PEKS encryption is probabilistic, identical keywords generate different ciphertext representations, which prevents inference attacks depending on keyword repetition patterns. Hence, proposed PT-CPABKS enables secure ranked retrieval while preserving keyword privacy, encrypted index confidentiality, trapdoor unlinkability, and resistance against statistical inference attacks depending on keyword repetition patterns. Hence, proposed PT-CPABKS enables secure ranked retrieval while preserving keyword produce different ciphertext representations, which prevents inference attacks depending on keyword repetition patterns. Hence, proposed PT-CPABKS enables secure ranked retrieval while preserving keyword privacy, encrypted index confidentiality, trapdoor unlinkability, and resistance against statistical inference attacks in cloud environments. Traditional CP-ABE models require repeated bilinear pairing and attribute-policy verification for each search and decryption operation, resulting in complexity of $O(n \cdot p + k \cdot p)$, where n is

the number of attributes, k is the number of query keywords, and p represents bilinear pairing cost. The MKSABE schemes include the complexity of $O(k \cdot n \cdot p)$ because both keyword matching and policy evaluation during query processing. The proposed PT-CPABKS introduces a policy-tiered index that separates searchable authorization from ciphertext decryption authorization by minimizing unnecessary attribute-policy evaluations and redundant bilinear pairing operations. The search complexity is minimized to $(O(n_s \cdot p + k))$, where $(n_s < n_f)$, effectively enhancing scalability and minimizing computational overhead in cloud environments.

3.4.3. Algorithm 1. Proposed PT-CPABKS method

Algorithm 1. Process of the PT-CPABKS

Actors: Key Generation Centre (KGC), Data Owner (DO), Cloud Server (CS), User (U_i)

System Setup (KGC)

Input: Security parameter λ

Output: Public parameters (PK), Master Secret Key (MSK)

KGC runs setup (λ) $\rightarrow$ (PK, MSK)

KGC distributes attribute-based secret keys (SK_i) to every user U_i

Data Encryption (DO)

Input: Document D , Keywords $\{w_1, w_2, \dots, w_n\}$, Access Policy P

Output: Encrypted document C , Encrypted Key C_k , Encrypted Index I

Generate random symmetric key K

$C \leftarrow \text{AES} - \text{GCM}.\text{Enc}(K, D)$

$C_k \leftarrow \text{CP} - \text{ABE}.\text{Enc}(\text{PK}, P, K)$

For every keyword w_i in $\{w_1, \dots, w_n\}$

$C_i \leftarrow \text{PEKS}.\text{Enc}(\text{PK}, w_i)$

Create policy-tiered index I using $\{C_i, P\}$

Upload $\{C, C_k, I\}$ to CS

Search Phase (User U_i)

Input: Keyword query q , user secret key SK_i

Output: Matching encrypted documents

U_i generates trapdoor $T_q \leftarrow \text{PEKS}.\text{Tradoor}(SK_i, q)$

Send T_q to CS

CS process Search (I, T_q)

CS returns encrypted results $\{C, C_k\}$ which match q

Decryption phase (User U_i)

Input: Encrypted Document C , Encrypted Key C_k, SK_i

Output: Plaintext Document D

$K' \leftarrow \text{CP-ABE.Dec}(SK_i, C_k)$

$D \leftarrow \text{AES-GCM.Dec}(K', C)$

Return plaintext Document D

4. Experimental analysis

The proposed model is simulated in Python 3.7, with the supporting cryptographic process combined from standard libraries. All experiments are conducted using the CloudSim tool, which acts as a client and server. The system was equipped with an Intel Core i7 processor running at 3.6 GHz and 16 GB RAM. The cryptographic runtime was measured in Python, and CloudSim was used separately for cloud communication simulation. To evaluate searchable encryption performance, the document collection includes 10,000 text files is considered with document sizes varying from 50 KB to 5 MB. The searchable vocabulary includes nearly 5000 unique keywords, where every document includes 15-40 searchable keywords. Multi-keyword query workloads include 1-5 keywords generated, and 2000 randomized search queries are executed to evaluate retrieval efficacy and scalability. The experimental environments include 500 users and 100 different attributes for CP-ABE policy generation. The access policies varied from 10 up to 100 attributes to validate policy complexity and multi-user scalability. The searchable index includes encrypted keywords, encrypted TF-IDF relevance scores, searchable access policies, and file-level decryption policies. The encrypted files are stored using IPFS, where each encrypted document generates a unique content hash for secure retrieval. The Encryption time (ms), decryption time (ms), search time (ms), latency (ms), Memory Usage (MB), Communication Cost (KB), Key Generation Time (ms), and execution time (ms) are the performance metrics used to validate the proposed model.

4.1. Quantitative analysis

Table 1 presents the encryption time (ms) for the proposed PT-CPABKS method as the number of attributes increases from 10 up to 100. Searchable Symmetric Encryption (SSE) incurs a lower cost because it relies on a lightweight symmetric process, whereas Public Key Encryption with Keyword Search (PEKS) incurs slightly higher but lower overhead due to the public-key process of keyword encryption, which is highly independent of attribute counts. CP-ABE incurs growth because of pairing computations that scale linearly with the number of attributes in the policy. Multi-keyword searchable retrieval over Encrypted Data (MRSE) incurs

moderate cost and represents the extra task required for multi-keyword ranking. The proposed PT-CPABKS consistently outperforms traditional CP-ABE in terms of encryption time, thereby reducing cost and maintaining dual access control using file- and index-level policies.

Table 1. Encryption time (ms) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	12	8	35	25	30
20	14	9	55	38	42
30	15	10	78	52	60
40	16	12	105	70	80
50	17	13	135	92	105
60	19	15	168	118	132
70	20	16	205	148	160
80	21	18	245	182	195
90	22	19	288	220	230
100	23	20	335	262	270

Table 2 presents the decryption time of the proposed method with traditional models as the number of attributes increases from 10 up to 100. SSE provides quick performance because it requires only a lightweight symmetric process, whereas PEKS adds negligible overhead as it mainly handles keyword matching rather than bulk decryption. The proposed PT-CPABKS obtains less decryption time than traditional models, thereby minimizing the redundant attribute process using its policy-tiered index mechanism.

Table 2. Decryption time (ms) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	5	4	28	12	10
20	6	5	45	20	16
30	6	5	68	30	24
40	7	6	95	42	34
50	8	6	125	56	46
60	9	7	158	72	60
70	9	8	195	90	76
80	10	8	235	110	94
90	11	9	278	132	114
100	12	10	325	156	136

Table 3. Search time (ms) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	15	5	40	25	12
20	16	6	65	38	18
30	18	6	95	55	26
40	20	7	130	75	36
50	22	7	170	98	48
60	24	8	215	124	62
70	25	8	265	153	78
80	27	9	320	185	96
90	28	9	380	220	116
100	30	10	445	258	138

Table 3 presents the search time performance of the proposed PT-CPABKS as the attributed number varies from 10 up to 100. The proposed PT-CPABKS effectively enhances the efficacy of CP-ABE with search times because its policy-tiered index ensures fast filtering and avoids redundant attributes. Although slower than SSE and PEKS, the proposed model provides a balanced trade-off between performance and fine-grained access control, making it suitable for real-world multi-user search scenarios.

Table 4 presents the latency performance of the proposed PT-CPABKS model as the number of attributes increased. The proposed PT-CPABKS effectively minimizes query latency compared to CP-ABE because it utilizes a policy-tiered index that filters users and attributes, minimizing redundant policy evaluations. When slower than SSE and PEKS, the proposed model provides a strong balance by minimizing the latency related to CP-ABE and MRSE while maintaining fine-grained, dual-policy access control. Table 5 shows the memory usage of the proposed model with traditional algorithms as the number of attributes increases. The proposed PT-CPABKS minimizes this overhead in memory usage by separating file-level and index-level policies, thereby avoiding redundant attribute-based ciphertext elements. Although the proposed model is slightly higher than SSE and PEKS, it is more effective than the traditional CP-ABE, which enables fine-grained, dual-policy access control.

Table 4. Latency (ms) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	22	8	55	32	18
20	24	9	85	50	26
30	26	10	120	72	36
40	28	11	160	96	48
50	30	12	205	123	62
60	32	13	255	152	78
70	34	14	310	184	96
80	36	15	370	218	116
90	38	16	435	254	138
100	40	17	505	292	162

Table 5. Memory Usage (MB) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	18	12	40	28	24
20	20	14	65	42	34
30	22	16	95	58	46
40	24	18	130	76	60
50	26	20	170	96	76
60	28	22	215	118	94
70	30	24	265	142	114
80	32	26	320	168	136
90	34	28	380	196	160
100	36	30	445	226	186

Table 6 represents the communication cost (KB) of the data exchanged per file across the number of attributes. SSE consistently incurs a lower cost because it transmits only compact symmetric ciphertexts and index pointers, whereas PEKS incurs a slightly higher overhead because of the transmission of encrypted keyword tags in addition to the symmetric ciphertext. The proposed PT-CPABKS minimizes the overhead with a low communication cost. This enhancement is obtained by its policy-tiered mechanism, which separates and supports secure keyword searches. Table 7 shows key generation time for the proposed PT-CPABKS as the number of attributes increases. SSE needs less time because it relies on one symmetric key, which is lightweight to generate, whereas PEKS is slightly costlier because of the requirement of asymmetric key pair generation, though it remains constant and independent of the number of attributes. The proposed PT-CPABKS minimizes this overhead in key generation times through optimizing attribute handling by a policy-tiered architecture, which avoids redundant per-attribute computations.

Table 6. Communication cost (KB) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	22	10	180	45	68
20	23	10	260	48	92
30	24	11	350	52	120
40	25	11	450	56	150
50	26	12	560	60	185
60	27	12	680	64	222
70	28	13	810	68	260
80	29	13	950	72	305
90	30	14	1100	76	355
100	31	14	1250	80	410

Table 7. Key Generation time (ms) vs Number of attributes

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	8	5	32	10	18
20	9	5	60	14	28
30	10	6	92	18	40
40	11	6	128	22	54
50	12	7	168	26	70
60	13	7	212	30	88
70	14	8	260	34	108
80	15	8	312	38	130
90	16	9	368	42	154
100	17	9	428	46	180

Table 8. Encryption, search and decryption execution time

Number of attributes	PEKS	SSE	CP-ABE	MRSE	Proposed PT-CPABKS
10	32	17	103	62	52
20	36	20	165	96	76
30	39	21	241	137	110
40	43	25	330	187	150
50	47	26	430	246	199
60	52	30	541	314	254
70	54	32	665	391	314
80	58	35	800	477	385
90	61	37	946	572	460
100	65	40	1105	676	544

Table 8 represents the overall execution time, which is calculated as the cumulative cost of encryption, search, and decryption for the proposed PT-CPABKS as the number of attributes increases. The proposed PT-CPABKS outperforms CP-ABE in terms of execution time. The reported 544 ms represents the sum of encryption, search, and decryption time at 100 attributes, excluding key generation, IPFS retrieval, and network latency. This minimization is attributed to a policy-tiered mechanism that offloads part of the policy enforcement to a searchable index and removes redundant per-attribute operations.

4.2. Comparative analysis

Table 9 evaluates the efficacy of the proposed PT-CPABKS model by conducting a comparative analysis against existing algorithms such as S3DBMS [21] (Table 9) and LFD-PEKS [22] (Tables 10 and 11). Compared to S3DBMS [21], the proposed model obtains a reduction in document encryption time against LFD-PEKS [22], resulting in less running time while scaling with the number of keywords. Similarly, in key generation, the proposed PT-CPABKS maintains lightweight overhead compared to ABCSE-MU [24], which increases. Compared with blockchain-based ABSE [25], the proposed model requires less key generation and encryption time during the search time despite its strong dual-policy enforcement. Finally, compared with ABSE [26], the proposed model reduces the encryption and search times, thereby demonstrating its scalability and suitability for large-scale cloud applications.

Table 9. Comparative analysis of the proposed model with S3DBMS [21]

Method	Number of attributes	EncDoc time (ms)
S3DBMS [21]	5	220
	10	350
	15	540
	20	710
	25	900
Proposed PT-CPABKS	5	20
	10	30
	15	48
	20	66
	25	88

Table 10. Comparative analysis of the proposed model with LFD-PEKS [22] with number of users

Method	Number of users	Running time (ms)
LFD-PEKS [22]	10	220
	20	420
	30	610
	40	825
	50	980
Proposed PT-CPABKS	10	70
	20	135
	30	200
	40	275
	50	330

These outcomes show that the PT-CPABKS balances by minimizing computational and communication overhead while providing fine-grained, multi-user access control with secure keyword search.

Table 11. Comparative analysis of the proposed model with LFD-PEKS [22] with number of keywords

Method	Number of keywords	Running time (ms)
LFD-PEKS [22]	100	6000
	200	11000
	300	16500
	400	22000
	500	27500
Proposed PT-CPABKS	100	2100
	200	3900
	300	5800
	400	7700
	500	9600

Table 12. Comparative analysis of proposed model with ABCSE-MU [24]

Method	Number of attributes	Key generation time (ms)
ABCSE-MU [24]	15	100
	20	200
	25	500
	30	600
	35	620
	40	820
Proposed PT-CPABKS	15	23
	20	28
	25	34
	30	40
	35	47
	40	54

Table 13. Comparative analysis of proposed model with Novel access control based on blockchain and ABSE [25]

Method	Number of attributes	Key generation time (ms)	Encryption time (ms)	Search time (ms)
Novel access control based on blockchain and ABSE [25]	10	30	30	10
	20	50	55	10
	30	70	65	10
	40	130	120	10
	50	150	140	10
Proposed PT-CPABKS	10	18	30	12
	20	28	42	18
	30	40	60	26
	40	54	80	36
	50	70	105	48

4.3. Research implications

The findings of the proposed PT-CPABKS model have several essential implications, which are described below.

- Advancement in Secure Cloud Search – it determines how integrating CP-ABE with a policy-tiered index efficiently balances fine-grained access control and system efficacy.

Scalability for Multi-User environments: This shows that optimized attribute handling minimizes overhead and makes large-scale deployments feasible.

- It provides extensive performance evaluations across multiple metrics, such as encryption, decryption, search, latency, communication, and memory, and offers effective searchable encryption.

4.4. Limitations

Although the proposed PT-CPABKS model improves efficacy and scalability, it still has limitations. The proposed methods introduce a slightly higher search overhead than lightweight symmetric models because of their dual-policy enforcement. However, the policy-tiered index minimizes the cryptographic process, as it requires additional storage management for large and dynamic attributes. In addition, the IPFS upload/retrieval latency and network bandwidth sensitivity were not separately reported.

5. Conclusion and future work

This manuscript proposes a PT-CPABKS model that addresses the performance and security drawbacks of existing searchable encryption algorithms. By integrating CP-ABE, AES-GCM, and PEKS with a policy-tiered index, the proposed model achieves an efficient balance between efficacy and fine-grained access control. The proposed PT-CPABKS method is processed by integrating CP-ABE for fine-grained access control, AES-GCM for effective file encryption, and PEKS for secure keyword search while incorporating a policy-tiered index, which reduces redundant attribute processing and accelerates query processing. This integration model enables high security and practical efficacy in multi-user cloud environments. Although the search costs are slightly higher than those of lightweight models, the proposed PT-CPABKS provides strong privacy and security under standard CP-ABE/PEKS assumptions. Overall, the proposed PT-CPABKS provides a scalable, secure, and effective model for multi-user cloud environments.

Future work will focus on addressing the drawbacks of PT-CPABKS by further optimizing the search efficacy using lightweight indexing and caching mechanisms.

References

1. Wu, Z., H. Wang, J. Wan, L. Zhang, J. Huang. An Inner Product Predicate-Based Medical Data-Sharing and Privacy Protection System. – IEEE Access, Vol. **12**, 2024, pp. 68680-68696.
2. Ghopur, D. Attribute-Based Searchable Encryption with Forward Security for Cloud-Assisted IoT. – IEEE Access, Vol. **12**, 2024, pp. 90840-90852.
3. Lin, J. K., W. T. Lin, J. L. Wu. Flexible and Efficient Multi-Keyword Ranked Searchable Attribute-Based Encryption Schemes. – Cryptography, Vol. **7**, 2023, No 2, 28.
4. Han, J., Z. Li, J. Liu, H. Wang, M. Xian, Y. Zhang, Y. Chen. Attribute-Based Access Control Meets Blockchain-Enabled Searchable Encryption: A Flexible and Privacy-Preserving Framework for Multi-User Search. – Electronics, Vol. **11**, 2022, No 16, 2536.
5. Zhu, X., P. Shen, Y. Dai, L. Xu, J. Hu. Privacy-Preserving and Trusted Keyword Search for Multi-Tenancy Cloud. – IEEE Transactions on Information Forensics and Security, Vol. **19**, 2024, pp. 4316-4330.
6. Xu, L., X. Cheng, W. Tian, H. Wang, Y. Zhang. Cloud-Assisted Privacy-Preserving Spectral Clustering Algorithm in a Multi-User Setting. – IEEE Access, Vol. **12**, 2024, pp. 75965-75982.
7. Sonkoly, B., B. G. Nagy, J. Dóka, Z. Kecskés-Solymosi, J. Czentye, B. Formanek, D. Jocha, B. P. Gerő. An Edge Cloud-Based Coordination Platform for Multi-User AR Applications. – Journal of Network and Systems Management, Vol. **32**, 2024, 40.
8. Han, H., Z. Wang, Z. Xu, X. Dong, W. Tian. Enhancing IoT Security and Efficiency: A Blockchain-Assisted Multi-Keyword Searchable Encryption Scheme. – IEEE Access, Vol. **12**, 2024, pp. 148677-148692.
9. Savard, C., N. Manganelli, B. Holzman, L. Gray, A. Perloff, K. Pedro, K. Stenson, K. Ulmer. Optimizing High-Throughput Inference on Graph Neural Networks at Shared Computing Facilities with the NVIDIA Triton Inference Server. – Computing and Software for Big Science, Vol. **8**, 2024, No 1, 14.
10. Du, Z., Y. Ni, H. Tao, M. Yin. Joint Optimization of Offloading Strategy and Resource Allocation for Multi-User in Dynamic Vehicular Edge Computing Systems. – Simulation Modelling Practice and Theory, Vol. **136**, 2024, 103001.
11. Liu, X. Y., L. J. Huan, W. J. Li, R. R. Sun. An Efficient Multi-Data Owner Cooperative Resource Sharing Scheme Against Key Regeneration in Edge Computing. – Journal of Systems Architecture, Vol. **146**, 2024, 103032.
12. Wang, F., H. Xiao, J. Wang, Y. Wang, C. Cao. Efficient Secure Channel-Free Identity-Based Searchable Encryption Schemes with Privacy Preserving for Cloud Storage Service. – Journal of Systems Architecture, Vol. **149**, 2024, 103089.
13. Chen, H., S. Tan, X. Ma, X. Lin, Y. Yao. Multi-Keyword Ranked Search on Encrypted Cloud Data Based on Snow Ablation Optimizer. – Symmetry, Vol. **17**, 2025, 1043.
14. Ling, Z., B. Wang. Updatable Multi-User Dynamic Searchable Encryption Scheme with Bidirectional Verification. – Mathematics, Vol. **13**, 2025, No 12, 1984.

15. Choi, H. J., N. Komuro, W. S. Kim. Microservices-Based Resource Provisioning for Multi-User Cloud VR in Edge Networks. – *Electronics*, Vol. **13**, 2024, No 15, 3077.
16. Dhamdhare, S. D., M. Sivakkumar, V. Subramanian. Cloud Data Security with Deep Maxout-Assisted Data Sanitization and Restoration Process. – *High-Confidence Computing*, Vol. **5**, 2025, No 1, 100238.
17. Zhao, M., L. Liu, Y. Ding, H. Deng, H. Liang, H. Wang, Y. Wang. Lightweight and Privacy-Preserving Multi-Keyword Search over Outsourced Data. – *Applied Sciences*, Vol. **13**, 2023, No 5, 2847.
18. Liu, G., H. Xie, W. Wang, H. Huang. A Secure and Efficient Electronic Medical Record Data Sharing Scheme Based on Blockchain and Proxy Re-Encryption. – *Journal of Cloud Computing*, Vol. **13**, 2023, 44.
19. Wang, S., N. Luo, B. Xing, Z. Sun, H. Zhang, C. Sun. Blockchain-Based Proxy Re-Encryption Access Control Method for Biological Risk Privacy Protection of Agricultural Products. – *Scientific Reports*, Vol. **14**, 2024, 20048.
20. Guo, K., Y. Han, R. Wu, K. Liu. CD-ABSE: Attribute-Based Searchable Encryption Scheme Supporting Cross-Domain Sharing on Blockchain. – *Wireless Communications and Mobile Computing*, Vol. **2022**, 2022, No 1, 6719302.
21. Khan, S., H. Abbas, M. Binsawad. Secure Semantic Search Using Deep Learning in a Blockchain-Assisted Multi-User Setting. – *Journal of Cloud Computing*, Vol. **13**, 2024, No 1, 29.
22. Zhang, S., Y. Yang, W. Liang, J. Zhou, K. Li, R. G. Crespo. LFD-PEKS: A Lightweight and Fast Dynamic Public-Key Encryption with Keyword Search Scheme for Multi-User Environments. – *Computer Standards & Interfaces*, Vol. **95**, 2026, 104050.
23. Tang, B., Y. H. Zhou, Y. G. Yang, B. Gong, N. Zhihong. A Lattice-Based Multi-Authority Updatable Searchable Encryption Scheme for Serverless Architecture with Scalable On-Demand Result Processing. – *Computer Standards & Interfaces*, Vol. **93**, 2025, 103956.
24. Zhao, F., C. Peng, D. Xu, Y. Liu, K. Niu, H. Tang. Attribute-Based Multi-User Collaborative Searchable Encryption in COVID-19. – *Computer Communications*, Vol. **205**, 2023, No 1, pp. 118-126.
25. Yan, L., L. Ge, Z. Wang, G. Zhang, J. Xu, Z. Hu. Access Control Scheme Based on Blockchain and Attribute-Based Searchable Encryption in Cloud Environment. – *Journal of Cloud Computing*, Vol. **12**, 2023, 61.
26. Feng, T., S. Miao, C. Liu, R. Ma. Verifiable Keyword Search Encryption Scheme That Supports Revocation of Attributes. – *Symmetry*, Vol. **15**, 2023, No 4, 914.
27. Shen, H., J. Zhou, G. Wu, M. Zhang. Multi-Keywords Searchable Attribute-Based Encryption with Verification and Attribute Revocation over Cloud Data. – *IEEE Access*, Vol. **11**, 2023, pp. 139715-139727.
28. Shiraly, D., Z. Eslami, N. Pakniat. Hierarchical Identity-Based Authenticated Encryption with Keyword Search over Encrypted Cloud Data. – *Journal of Cloud Computing*, Vol. **13**, 2024, No 1, 112.
29. Zhao, J., H. Huang, Y. Xu, X. Zhang, H. Du, C. Huang. Verifiable Attribute-Based Multi-Keyword Search Scheme with Sensitive Information Hiding for Cloud-Assisted e-Healthcare Sharing Systems. – *Theoretical Computer Science*, Vol. **1023**, 2025, 114895.

30. Xu, Z., C. Tian, G. Zhang, W. Tian, L. Han. Forward-Secure Multi-User and Verifiable Dynamic Searchable Encryption Scheme within a Zero-Trust Environment. – Future Generation Computer Systems, Vol. **166**, 2025, 107701.
31. Maathavan, K. S. K., S. Venkatraman. A Secure Encrypted Classified Electronic Healthcare Data for Public Cloud Environment. – Intelligent Automation & Soft Computing, Vol. **32**, 2022, pp. 765-779.
32. Karimunnisa, S., Y. Pachipala. Deep Learning-Driven Workload Prediction and Optimization for Load Balancing in Cloud Computing Environment. – Cybernetics and Information Technologies, Vol. **24**, 2024, No 3, pp. 21-38.
33. Ilankumaran, A., S. J. Narayanan. An Energy-Aware QoS Load Balance Scheduling Using Hybrid Gaaco Algorithm for Cloud. – Cybernetics and Information Technologies, Vol. **23**, 2023, No 1, pp. 161-177.

*Fast-track. Received: 07.04.2026; First Revision: 18.06.2026;
Second Revision: 21.06.2026; Accepted: 25.06..2026*

© 2026 by the authors. Licensee IICT-BAS, Sofia, Bulgaria. This article is an open access article distributed under the terms and conditions of the **Creative Commons Attribution-NonCommercial-NoDerivativs** license.