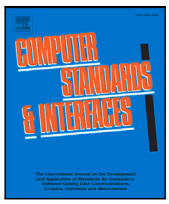
Computer Standards & Interfaces 89 (2024) 103824

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi



BMSE: Blockchain-based multi-keyword searchable encryption for electronic medical records☆

Shen Fanfan [a], Shi Lin [a,*], Zhang Jun [b], Xu Chao [a], Chen Yong [a], He Yanxiang [c]

[a] College of Computer Science, Nanjing Audit University, Nanjing 211815, China
[b] College of Software, East China University of Science and Technology, Nanchang 330013, China
[c] College of Computer Science, Wuhan University, Wuhan 430072, China

ARTICLE INFO

Keywords:
Blockchain
Multiple keyword
Searchable encryption
K-means

ABSTRACT

The storage of electronic medical records (EMRs) is an area of extensive research, and healthcare systems often delegate this task to cloud service providers (CSP). Typically, CSP transmits the encrypted EMRs to a cloud server with a searchable encryption scheme for easy retrieval. However, the enormous power held by centralized CSP poses a potential threat to patients' personal privacy, as it can lead to unauthorized access and misuse of medical data by both CSP and data users, such as doctors. This paper proposes a blockchain-based multi-keyword searchable encryption (BMSE) electronic medical record solution. The scheme consists of two parts. On the one hand, our solution involves the integration of blockchain technology and the utilization of advanced encryption standard (AES) for symmetric data encryption. Additionally, we employ attribute-based encryption (ABE) to encrypt the search index. This approach aims to address the issue of excessive power held by centralized CSP, which can potentially result in the compromise of patients' privacy. On the other hand, we use the K-means algorithm to cluster the documents, and use the relevance score of keywords and documents as the search index to solve the problem of low efficiency of the existing multi-keyword searchable encryption schemes. Finally, we verify the safety of BMSE through safety analysis, and the experimental analysis shows that BMSE improves the search efficiency.

1. Introduction

Electronic medical records (EMRs) are a digital way of creating, storing and using information about an individual's health status. At the patient's subsequent appointment, EMRs can record more precise information on the patient's diagnosis, the doctor will be able to accurately characterize the patient's prior medical state. For this reason, the doctor can make a more accurate diagnosis. Healthcare systems around the world have gradually established their own information systems to store EMRs. But as the number of visits rises, so does the number of EMRs. If we continue to store these EMRs on the hospital's local servers, On the one hand, it puts a serious burden on the hospital's servers, and on the other hand, it prevents data from being shared between them [1].

To solve this problem, a lot of hospitals have begun to utilize the cloud servers [2], enabling patients to transfer their EMRs to cloud servers, This not only solves the storage problem of EMRs in hospitals, but also enhances data sharing. However, cloud service providers (CSP) are not always reliable, and some unscrupulous CSP may sell patients' electronic medical records to third parties for profit, compromising the privacy of the patients. In order to ensure the patients' privacy security, the electronic medical records of patients are encrypted before being transferred to a cloud server. Encryptable search technology can be used to facilitate the retrieval of encrypted data when EMRs need to be downloaded, making EMRs easier to utilize. It is important to note that centralized CSP hold a large number of medical indexes, which could lead to CSP colluding with doctors and other researchers to steal and profit from patients' medical data. In addition, centralized CSP are at risk of security breaches, which could potentially tamper with the results of calculations stored and performed on EMRs once an attacker illegally gains access. For example, an attacker may use a distributed denial-of-service attack to bring down the entire system with the intention of accessing the data for free. These problems are

☆ This work is supported by the National Natural Science Foundation of China (61902189, 71972102, 62162002), the Basic Science (Natural Science) Research Project of Colleges and Universities in Jiangsu Province (22KJA520004), the Natural Science Foundation of Jiangxi Province (20212BAB202002), and Postgraduate Research & Practice Innovation Program of Jiangsu Province (SJCX22_1000).

* Corresponding author.
E-mail address: mz2109103@stu.nau.edu.cn (L. Shi).

https://doi.org/10.1016/j.csi.2023.103824
Received 2 August 2023; Received in revised form 2 November 2023; Accepted 13 December 2023
Available online 19 December 2023

rooted in the fact that centralized CSP have too much power in the absence of effective oversight. Therefore, there is an urgent need to implement more secure searchable encryption schemes to address these issues [3].

In order to solve this problem, C. Esposito et al. applied blockchain technology to the medical field, uploading patients' EMRs to the blockchain to achieve data access and sharing [4]. As a distributed public ledger, blockchain has the characteristics of decentralization and traceability, and using blockchain to store EMRs can better protect the privacy of patients [5]. Zheng et al. had applied smart contracts and Ethereum to store and access data, guaranteeing the confidentiality of data [6]. Ethernet is a reliable and decentralized blockchain platform. Ethereum's smart contracts can automatically execute contracts to prevent malicious actors from interfering with contract execution [7]. In addition, once the smart contract is installed on Ethereum, its code cannot be changed, and the smart contract prevents hackers from modifying it for financial gain. The current research mainly has two problems. On the one hand, data is stored in centralized CSP, which will give centralized CSP too much power to operate patients' EMRs, which will lead to the disclosure of patients' privacy. On the other hand, although some existing methods use multi-keyword to search the data, they still have some shortcomings in terms of search time.

Considering the issues in current searchable encryption schemes, this study proposes a blockchain-based multi-keyword searchable encryption scheme for electronic medical records (BMSE). By introducing blockchain technology, BMSE can effectively avoid the malicious impact of centralized CSP and enable patients and data users to conduct fair and impartial transactions. By using Simhash values and K-means clustering, documents can be processed in advance, which can significantly reduce the overhead of retrieving data for patients and data consumers. In the scheme, patients can search and decrypt their own EMRs, and other authorized users can access the data without violating patient privacy. The following are the main contributions:

- We propose a novel blockchain-based searchable encryption solution for healthcare systems with a smart contracts secure protocol, The scheme ensures that data users can obtain the electronic medical records they need for a fee, while also ensuring that patients receive appropriate compensation.
- We employ K-means and topic extraction methods for EMRs. The size of the search set is decreased, thereby increasing the search's effectiveness.
- The evaluation results demonstrate that BMSE can really shorten search times while maintaining search accuracy, and the security study demonstrates that BMSE is secure.

The paper is structured as follows, Section 2 reviews related work, Section 3 gives some theorems required in the scheme, Section 4 presents the system model, Section 5 focuses on the construction of the BMSE scheme, Section 6 proves the safety of the scheme, Section 7 evaluates the efficiency of the scheme, and Section 8 gives the conclusions of the paper.

2. Related work

In recent years, searchable encryption has become a key technology in cloud storage services. Here, we discuss the connection and difference between our scheme and related works from two aspects of encryption and search.

2.1. Encrypt

After extensive research, Song et al. first used a new technique for remote searching of encrypted data using untrusted servers in 2000 and provided a proof of security for the generated encryption system [8]. However, Song et al.'s method only allows a specific user holding a private key to encrypt and search the data, in order to solve this problem, Boneh et al. proposed a public key searchable encryption scheme [9]. In 2005, SaHai et al. used attribute encryption in encryption and proposed a fine-grained attribute-based encryption scheme (ABE), where the data owner can formulate attribute-based Boolean formulas and only users who satisfy the attributes can decrypt them [10]. In 2014, Sun et al. proposed the first multi-keyword searchable scheme inspired by attribute-giving encryption (ABE) [11]. Wang et al. in 2019 proposed a secure search scheme based on combining keyword search and proxy re-encryption for data sharing between different healthcare organizations [12]. This scheme has efficient user revocation and scalable fine-grained search authorization. Liu et al. in 2019 proposed a blockchain-based privacy-preserving data sharing scheme for EMRs, in which the raw data of EMRs is securely stored in the cloud while their indexes are retained in a federated blockchain with tamper-proof features [13]. In 2021, Tahir et al. proposed a blockchain-based contact tracking encryption scheme that encrypts all data of a contact and stores it on the blockchain by using homomorphic encryption, which can be very protective of personal privacy [14].

AES has been widely used in a variety of security constrained applications, but many applications are limited by power and resources, so it needs reliable and efficient hardware implementation. Kermani studied AES-GCM mode in 2011 and found that nonlinear conversion is the key to achieve low hardware complexity and low power consumption. They proposed a parallel, high-performance solution to improve throughput and achieve low latency on GCM hardware. Finally, they tested the proposed AES-GCM architecture and fault detection method on application-specific integrated circuit (ASIC) and field-programmable gate array (FPGA) platforms, and proved that the proposed architecture can provide higher efficiency and more reliable fault detection ability [15]. In 2015, Kermani conducted research in the emerging field of secure deep embedded systems, and proposed an effective research and education integration strategy, which verified the possibility of the integration of teaching and research in the co-design of software and hardware security systems [16]. Sarker et al. proposed a simple and flexible encryption scheme SABER in 2022, which can well solve potential attacks in the post-quantum era. This scheme can be executed in hardware alone or hardware/software co-processing. In addition, they also proposed a compact signature scheme called Falcon, which is proved to be a very suitable signature algorithm for PQC [17]. In 2023, Sarker et al. proposed an efficient polynomial multiplication technique NTT based on lattice-based post-quantum cryptography. By embedding a new error detection scheme in it, it can detect temporary and permanent faults and is suitable for various restricted usage patterns [18].

Cryptographic architectures provide different security properties to protect sensitive usage models, but security is not guaranteed. Therefore, Mozaffari Kermani proposed an error detection method for the camellia block cipher in 2016, taking into account both linear and nonlinear subblocks. A fault detection architecture is proposed to achieve the goal of security and reliability [19]. In the same year, they did a study on the emerging embedded system and proposed to re-examine the traditional security mechanism in order to deal with more malicious attacks. In addition, they believed that the lightweight encryption system had great potential in the emerging security method [20]. The intersection between lightweight cryptography (LWC) and advanced Artificial intelligence (AI) language models was first explored by Cintas-Canto et al. in 2023. ASCON is increasingly important for protecting data security, and OpenAI's large language model (LLM) has great potential for generating complex human-like text. They proposed a new scheme to implement ASCON of NIST LWC standard using GPT-4 model, which opens up ideas for the application of AI language models in cryptography [21].

Our scheme is a blockchain-based encryption scheme that encrypts EMRs using the Advanced Encryption Standard (AES), and encrypts the keyword indexes using Attribute-Based Searchable Encryption (ABSE) before storing them on the blockchain, which can be a good way to protect users' privacy.

2.2. Side channel attacks

Azarderakhsh et al. presented the first implementation of the supersingular homology Diffie–Hellman (SIDH) key exchange in 2016, which was the first hardware implementation based on homology cryptography. This implementation can increase speed by minimizing pipeline stalls and can be implemented on reconfigurable hardware [22]. In order to evaluate the resistance of CRYSTALS-Kyber to side-channel attacks, Dubrova et al. in 2023, proposed a message recovery attack based on deep learning, mainly by using recursive learning, a new neural network training method that can achieve 99% probability of reply to message bits from higher-order masks [23]. In 2023, Kaur et al. made the first survey on the current work of lightweight cryptography standards, summarized the implementation of ASCON cipher on different hardware platforms, analyzed the differential and side channel analysis attacks on ASCON cipher suites and the countermeasures, and provided their own insights and prospects [24]. Canto, A.C et al. investigated the emerging security problems in the post-quantum era in 2023. Although the rapid development of quantum computing can bring opportunities for the progress of science and technology, there will also be security problems. Many encryption algorithms are vulnerable to side channel attacks, so it is necessary to continue to do research on solving these problems and propose countermeasures to solve side channel attacks in the future [25].

2.3. Search

Research on searchable encryption in retrieving data is also evolving. In 2018, Liu et al. implemented a multi-keyword query with a two-keyword index structure to improve search efficiency [26]. In 2016, Gagan et al. clustered similar documents together by keyword similarity and sent the hash of the calculated keywords to the data owner [27], allowing comparison of clusters during retrieval, the hash value of the index and the hash value of the trapdoor to find the cluster, which can reduce the problem of excessive overhead of trapdoor and index comparison operations, but the shortcoming is that the weight of keywords in the document and the distribution of clusters are neglected. Zhu et al. used K-means clustering algorithm to cluster the documents and improve the similarity of documents within the same cluster [28]. Shu et al. proposed an efficient non-interactive, agentless cryptographic search matching method, but the drawback is that it does not support multi-keyword search [29]. In 2022, Zheng et al. proposed a blockchain-based multi-keyword searchable encryption scheme for COVID-19 contact tracking data, which combines ABE and blockchain technologies to ensure the integrity of the search index of data collections [6].

Our scheme is a multi-keyword retrieval based scheme that clusters documents with K-means so that documents with high similarity are in the same cluster, and then generates indexes for each cluster to improve the search efficiency.

3. Preliminaries

This section briefly introduces the relevant theoretical knowledge of prime order bilinear groups, Decisional Bilinear Diffie–Hellman (DBDH) Assumptions, Decisional Diffie–Hellman (DDH) assumption.

3.1. Bilinear mapping

Theorem 1. *Bilinear mapping:*

The prime bilinear group can be described by a quintuple (p, G_1, G_2, G_T, e). *Where* p *is a large prime associated with a given safety constant* λ, G_1, G_2, G_T *are all multiplicative cyclic groups of order* p. *Define a bilinear map* $e: G_1 \times G_1 \rightarrow G_2$, *then it satisfies the following properties [30]:*

- *Bilinear: for arbitrary* $x \in G_1, y \in G_2, a, b \in Z_p$, *it holds that* $e(x^a, y^b) = e(x, y)^{ab}$.
- *Non-degenerative: presence* $x \in G_1, y \in G_2$, *it holes that* $e(x, y) \neq 1$.
- *Computability: for arbitrary* $x \in G_1, y \in G_2$, *existence of an efficient algorithm to calculate* $e(x, y)$.

Table 1
Symbolic meaning.

Symbolic	Descriptions
$attr_i$	Attributes of user i
v_i	Attribute values for user i
F	Document collection
FS	Simhash value set of the document
L	Clustered document set
W	Keyword set
In	Index

3.2. DBDH assumptions

Theorem 2. *DBDH Assumptions:*

- *Given the multiplicative groups* G_1 *and* G_2 *of order* p.
- *Randomly select the generator* $g \in G1$ *and the random numbers* $c_1, c_2, c_3 \in Z_p$.
- *Issue* $g, g^{c_1}, g^{c_2}, g^{c_3}, e(g,g)^{c_1c_2c_3}$ *and* $T \in G_2$ *to* $\mathcal{A}$.
- $e(g,g)^{c_1c_2c_3}$ *determines whether* T *is equal to* $e(g,g)^{c_1c_2c_3}$. *If it is equal,* $\mathcal{A}$ *outputs 1, otherwise outputs 0.*

The advantages of defining algorithm $\mathcal{A}$ *to solve the above problems are shown in formula (1):*

$$\begin{aligned} \mathrm{Adv}^{\mathrm{DBDH}} = &|\Pr[\mathcal{A}(g, g^{c_1}, g^{c_2}, g^{c_3}, e(g,g)^{c_1c_2c_3}) = 1] \\ &- \Pr[\mathcal{A}(g, g^{c_1}, g^{c_2}, g^{c_3}, T) = 1]|. \end{aligned} \tag{1}$$

If no polynomial-time algorithm solves the DBDH hypothesis by a non-negligible advantage, then we say that the DBDH hypothesis holds in groups G_1, G_2.

3.3. DDH assumptions

Theorem 3. *DDH Assumptions:*

The DDH hypothesis refers to the fact that it is difficult to distinguish between tuples (g, g^a, g^b, g^{ab}) *and* (g, g^a, g^b, g^z), *where* g *is the generator and* x, y, z *are random. Let* G *be a group of large prime* q *of order, and let* g *be the generator of* G, $x, y, z \leftarrow_R Z_q$, *then the following distributions are computationally indistinguishable:*

- *A random quadruple* $R = (g, g^x, g^y, g^z) \in G_q$.
- *DDH quadruple* $D = (g, g^x, g^y, g^{xy}) \in G_q$.

is computationally indistinguishable and is called the DDH hypothesis. Specifically speaking of an opponent A, As shown in formula (2), A distinguishes the advantages of R and D:

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{DDH}}(\kappa) = |\Pr[\mathcal{A}(R) = 1] - \Pr[\mathcal{A}(D) = 1]| < \varepsilon \tag{2}$$

where κ *is the safety parameter and* ε *is negligible.*

4. System model

The model of the data searchable encryption scheme for sharing data of electronic medical records under the federated chain is shown in Fig. 1.

The system model mainly includes six parts: Attribute Authorization Center (AAC), Cloud Service Provider (CSP), Doctor, Patient (PA), Data User (DU), Private Chain and Alliance Chain. The meaning of the individual letter symbols used in the programme is described in Table 1. A brief description of the system model is as follows:

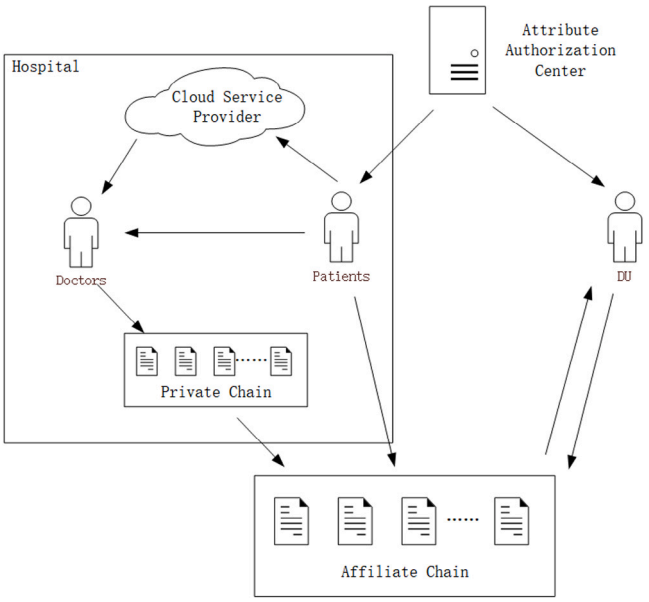


Fig. 1. System model.

- Attribute Authorization Center (AAC): AAC is responsible for generating and distributing public and private keys for all system users.
- Patient (PA): PA is the entity that owns the EMRs, owns the original content of the data, encrypts the document, encrypts the search index, and uploads it to the blockchain.
- Data User (DU): DU is the entity that requests encrypted data from the blockchain. Different DU has different attributes, and when they request data, they will return different data according to different attributes, and then decrypt it using the private key.
- Private Chain and Alliance Chain: Both the consortium chain and the private chain are components of the blockchain, which is responsible for storing encrypted data and returning the files required by the PA or DU according to the smart contract.

In addition, both PA and DU use smart contracts when interacting with encrypted data, PA mainly uploads indexes to smart contracts, and DU is to retrieve the required data from smart contracts.

The main workflow of the system is shown in Fig. 2 as the timing diagram, the main steps in the sequence diagram are as follows:

1. AAC is a trusted entity responsible for generating and sending public and private keys for all users.
2. PA processes the document and generates a keyword index and encrypts it, then uploads the data to the CSP for storage.
3. PA stores the encrypted index into smart contracts and deploys it on the blockchain for the purposes of fair payment and search.
4. DU generates search tokens, deployable smart contracts, sends them to the blockchain for transactions, and the search will only begin when a full fee is paid.
5. After the search is complete, the data is returned to DU, which then decrypts the file.

5. The proposed BMSE system

BMSE is a searchable encryption scheme based on blockchain and multi-keyword. In the scheme, PA uploads data to the blockchain and uploads the index to the smart contract for deployment. DU pays a certain fee when it needs to use the data, and then obtains the corresponding file based on its own attributes and uses the data through decryption. The specific implementation is as follows:

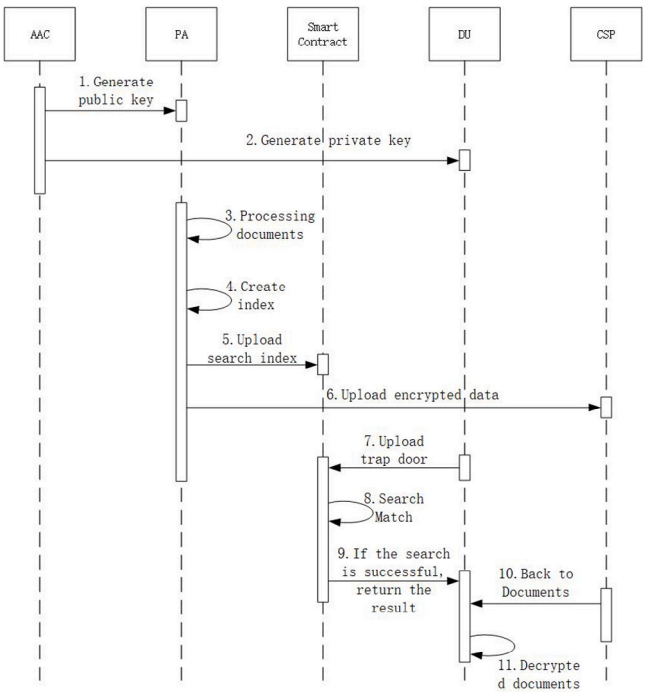


Fig. 2. Timing diagram.

5.1. Key generation

AAC initializes the system and gets all possible user attributes, $U = \{attr_1, attr_2, \ldots, attr_n\}$, where n is the size of the attribute set. For each attribute in the system, there are two values v_i and $\neg v_i$. The value of $\text{attr}_i = v_i$ when the data requester has the attribute attr_i and the value of $\text{attr}_i = \neg v_i$ when the data requester does not have the attribute attr_i. These values are used to determine whether the data requester's attribute contains the attribute. Then AAC starts generating the public key and the master private key. Given a bilinear group e: $G \times G \rightarrow G_T$, where p is the prime order of G and G_T, and One-item hash function H: $\{0, 1\}^* \rightarrow Z_p$. Then randomly pick the variable x, y, z and $\{u_1, u_2, \ldots, u_n\}$ in Z_p, pick the variable $\{t_1, t_2, \ldots, t_n\}$ in G. Afterwards, let $k_i = g^{u_i}$, $q_i = e(t_i, g)$. After all variables are set, the public key pk and the master private key msk are generated.

$$pk = \left(g, g^x, g^y, g^z, \left(k_i, q_i\right)\right) \tag{3}$$

$$msk = \left(x, y, z, \left(u_i, t_i\right)\right) \tag{4}$$

Then PA can then use the generated key to encrypt the data. The key generation algorithm is shown in Algorithm 1, in Algorithm 1, we achieve the purpose of prime mapping by importing the JPBC library, as a result, attribute-based keys can be generated, the public key is generated using the preset file provided by JPBC for prime mapping.

Algorithm 1 Key Generation Algorithm

Input: All possible user attributes
Output: Public key and master private key.
1: Initialize the JPBC library for the bilinear map.
2: Set new elements for each parameter.
3: **for** Each attribute **do**
4: Get the grouping of each attribute in the master private key.
5: Get the grouping of each attribute in the public key
6: **end for**
7: The multiplication is performed on each element

5.2. File processing

The experiments of various clustering algorithms in Ref. [31] shows that the search efficiency of K-means algorithm is higher than other clustering algorithms. Therefore, K-means algorithm is chosen to cluster documents in this scheme. The K-means algorithm is a divisive clustering algorithm that is suitable for most data. The algorithm has good scalability, is computationally simple and efficient, and has a low space complexity of linear complexity. Clustering is used to group documents into clusters before indexes are created, but clustering documents directly would take a lot of time, so we first calculate Simhash values for the documents and then cluster these Simhash values. To perform the clustering, the documents are first grouped into a dataset $F = \{f_1, f_2, \ldots, f_t\}$. The Simhash value $FS = \{fs_1, fs_2, \ldots, fs_t\}$ is then calculated for each document, and after the calculation is complete the K-means algorithm is applied on FS to generate k clusters obtain the set of similar documents $L = \{l_1, l_2, \ldots, l_k\}$. The specific execution process is as follows (Correspond to lines 1–15 of Algorithm 2):

1. Calculate the Simhash value for each document.
2. Randomly select k documents as the center of mass.
3. Calculate the Euclidean distance from each document to each center of mass.
4. Assign a document to its nearest cluster.
5. After the assignment, the center of mass is recalculated to see whether the clustering is convergent. If it is not, the above clustering steps are repeated.

For each document, keywords w can also be generated for each cluster in $L = \{l_1, l_2, \ldots, l_k\}$ by the latent Dirichlet distribution algorithm and added to the set of keywords $W_t = \{w_1, w_2, \ldots, w_k\}$, which will then be used as an index for each cluster. The main steps of using Latent Dirichlet Allocation (LDA) algorithm to generate keywords are as follows (Correspond to lines 16–25 of Algorithm 2):

1. The document set L is represented by the bag model.
2. Initializes the LDA model parameters: randomly initializes the document – topic distribution θ, and randomly initializes the topic – word distribution β.
3. Iteratively train the LDA model: For a given set of documents, the LDA model is iteratively trained. In each iteration, the topic and word distribution are updated until they converge.
4. Generate keyword: For a given document, infer its topic distribution. For each topic, select the words with the highest probability as keywords.

The document processing algorithm is shown in Algorithm 2:

Algorithm 2 File processing

Input: $F = \{f_1, f_2, \cdots, f_t\}$
Output: $L = \{l_1, l_2, \cdots, l_k\}$, $W_t = \{w_1, w_2, \cdots, w_k\}$
1: **for** $i = 1, i <= t, i++$ **do**
2: Calculate Simhash value fs_i
3: Get the set $FS = \{fs_1, fs_2, \cdots, fs_t\}$
4: **end for**
5: k points are randomly initialized as the starting centroid of the document.
6: **for** $i = 1; i < t; i++$ **do**
7: Compute the Euclidean distance from document i to the centroid
8: It is grouped into the cluster corresponding to the centroid with the smallest distance
9: **end for**
10: The mean of all the samples in the cluster is calculated to obtain the new centroid k'.
11: **if** $k = k'$ **then**
12: Output the set of documents after clustering.
13: **else**
14: Repeat steps 6-10.
15: **end if**
16: The document set L is represented by the bag model
17: randomly initializes the document-topic distribution θ, and randomly initializes the topic-word distribution β
18: **for** $l \in L$ **do**
19: **for** $d \in D$ **do**
20: The word frequency distribution of w is found from the word bag model
21: For each topic t, calculate the distribution of topic t in document d
22: For each topic t, Calculate the probability that the word w is generated by the topic t
23: **end for**
24: **end for**
25: Select the words with the highest probability as the keywords

5.3. Index generation

After the clustering is complete, an index is created for each cluster using the latent Dirichlet distribution algorithm and a keyword index is created for the documents within each cluster. This solution uses the indexing technique proposed in literature [32] to construct the keyword index, where a ranking function is used to calculate the relevance score of matching documents to a given search request. The main steps to generate a keyword index are as follows:

1. The data owner first creates a keyword index $In(w_q) = d(f_{kq})$ based on document set $F = \{f_1, f_2, \ldots, f_t\}$ and keyword set $W_t = \{w_1, w_2, \ldots, w_p\}$
2. The relevance score of the key set of document f_i is calculated by TF–IDF, and the calculation formula is shown in formula (5) [33]:

$$g_{ij} = \sum_{w \in w_q} \frac{1}{f_w} \bullet (1 + \ln F_{f,w}) \bullet \ln\left(1 + \frac{A}{F_w}\right) \tag{5}$$

3. Add the keyword score to the index to get the index $In(w_q) = d((f_{kq}) \parallel (g_{kq}))$ we need

The algorithm for index generation is shown in Algorithm 3:

Algorithm 3 Index generation

Input: $F = \{f_1, f_2, \cdots, f_t\}$, $W_t = \{w_1, w_2, \cdots, w_p\}$
Output: $In(w_q) = d((f_{kq}) \parallel (g_{kq}))$
1: Building keyword indexes $In(w_q) = d(f_{kq})$
2: Calculate the relevance score for the keyword set g_{kq}
3: Add the relevance score g_{kq} to $In(w_q) = d(f_{kq})$ to get the final index $In(w_q) = d((f_{kq}) \parallel (g_{kq}))$

5.4. File encryption

The data owner will first generate a secret key and then send it to the data requester. The key will be used in the Advanced Encryption Standard — Galois/Counter Mode (AES-GCM) [34] to encrypt and decrypt the document. The data owner can set a password during the encryption process, so the data requester also needs to know the password to decrypt the file. First, we initialize AES-GCM encryption library with Javax. Once initialized, the plaintext file is read into bytes for encryption purposes. A random salt and IV along with the password will then be generated and given to the data owner for encryption. If the encryption is successful, the encrypted file bytes will be written to a file and stored in the directory of that file. The keyword index

generated is encrypted using AES-GCM, with the difference that another key pre-installed in the system is used. This key is different from the key used for file encryption, and with the help of the password used in the encryption process, an attacker cannot easily encrypt/decrypt the file even if he can learn the key. Once the encryption is successful, the data is uploaded to the CSP and the keyword index is uploaded to the smart contract. The file encryption algorithm is shown in Algorithm 4:

Algorithm 4 File encryption

Input: cluster C, plain text M, keyword index In
Output: Encrypted cluster $C^{'}$, file $M^{'}$, keyword index $In^{'}$
initialize the JPBC library and determine the parameters.
convert the cluster, plaintext, and index to bytes.
generate random salt and IV.
encrypt the byte file with the generated salt and IV.
if encryption is successful **then**
    put the encrypted file in directory.
end if
Upload the data to the CSP and the keyword index to the smart contract.

5.5. Document search

When DU want to search for data, they first enter attributes and keyword indexes to generate search tokens. Then spend the full amount to set up a transaction, search through the smart contract, compare the keyword index, and if the corresponding keyword is retrieved, the corresponding document is returned by the CSP. When the search begins, the CSP selects the cluster that matches the keyword index, queries the relevance score for each document in the l_k cluster, and returns the former k documents to the user.

During the file search process, a B+ tree-based data structure is used to obtain the corresponding file list. Suppose the depth of the tree is i (where $i > 1$), indicating that the tree has multiple levels. Each file stores an index of its frequent keywords cumulatively in the tree at each level. The search process begins with the server comparing the query keywords to the first layer of each file. If a file matching the query keyword is found in the first layer comparison, the search process continues to the other layers of the tree. The layer-by-layer comparison makes the search process as efficient as unencrypted data. In the search phase, special emphasis is placed on top-k search. The server can process top-k searches as quickly as it can process plain text fields. This means that the server can quickly produce the k files that are most relevant to the query keyword to meet the needs of the user. By using the optimized data structure of B+ trees to organize and retrieve the list of files, this improved search method provides the ability to obtain the required files efficiently and accurately. It makes full use of the hierarchical characteristics of the tree structure, which makes the search process more efficient and scalable.

If the data is found, the CSP returns the data to DU and pays the PA. If no data is retrieved, the fee is returned to DU. The document search algorithm is shown in Algorithm 5:

5.6. File decryption

After initializing the JPBC library, combined with IV and salt, DU can then enter the key to upload the encrypted index and token to the smart contract. Once the index of the file is retrieved from the search results, the encrypted data can be downloaded and decrypted using AES-GCM. In this process, the decryption function will successfully execute to decrypt the encrypted file only if both inputs match the value used during encryption. If there is an error in either of them, then neither of them will successfully decrypt the file. The file decryption algorithm is shown in Algorithm 6:

Algorithm 5 Document search

Input: keyword indexing In, number of documents (k)
Output: top-k files
DU provide fees and keyword index In
Smart contracts open transactions
Start searching for documents
Gets the depth i of the tree
for $j = 1, j <= i, j++$ **do**
    Query keywords and compare them to the first layer of each file
    if Find a matching file **then**
        Continue to compare the keyword index with other levels of identification
    end if
    Refund the fee to DU
end for
Get all k documents that match

Algorithm 6 File decryption

Input: Ciphertext $M^{'}$,secret key s_k
Output: plain text M
initialize the JPBC library.
enter IV, salt and ciphertext.
reading the key.
upload the encrypted index and token to the smart contract.
if password is wrong **then**
    termination process
else
    decrypt the file using the key.
end if
output plaintext.

6. Security analysis

6.1. Confidentiality of data

Assuming that the attacker succeeds in gaining access to the encrypted file, he is also unable to read the dataset directly or perform common attacks on the encrypted dataset to recover the original file.

Proof. We encrypt the dataset using AES-GCM with a password chosen by the data owner. The encryption process is performed using a symmetric key and password generated by the data owner. Therefore, the file decryption operation must know these two values, the key and the password, in order to generate the original file. An attacker who does not know these values will not be able to decrypt the file and obtain any meaningful data. In addition, the operation method of AES-GCM encryption mode is more complex than other AES block cipher modes. Several XOR functions have been applied to a plaintext block before it is converted into a ciphertext block, and the pattern of the ciphertext block will be completely randomized with the participation of the IV. The attributes of the authentication tag will allow the detection of any modification of the data. Therefore, the confidentiality and integrity of the data are guaranteed.

6.2. The immutability of blockchain

The blockchain will decline to generate the transaction if the user is dishonest. As a result, the user cannot access the connected data file. The user suffers consequences as a result. If the data owner is dishonest, the blockchain will refuse to generate the transaction and the data owner will be penalized since the ciphertext and key encoded

Table 2
Functional analysis.

Functional	[9]	[12]	[13]	[33]	BMSE
Blockchain	✗	✗	✔	✗	✔
Privacy protection	✔	✔	✔	✔	✔
Multi-keyword	✗	✗	✗	✔	✔
Identification	✗	✔	✔	✔	✔

in the transaction are incorrect. If the server is dishonest, either he has delivered the incorrect result or the value he has embedded in the transaction is incorrect. The server is unable to obtain either the service fee or any unencrypted data in the first case. The user will receive accurate search results from the transaction if all three parties follow the protocol honestly and fairly, and the data owner and server will also receive the appropriate service fees.

6.3. Immutability and non-repudiation of data

If the search index is stored on a local server or the data is outsourced to the CSP, then the confidentiality and integrity of the data may be compromised. Since the CSP is untrusted, it is very possible for the CSP to view or modify the data we upload. Therefore, we use smart contracts as the search index and store the data in IPFS, so that we can identify and prevent outside attempts to modify the index and data we upload. Not only that, any access to data is also logged so that data requester cannot deny their transactions.

Proof. In the BMSE scheme, the search index will be uploaded to the blockchain, which in turn has immutability and distributed ledger. Anything we store must then be checked for validity with the network participants before any transactions can be made. Any attempt to tamper with data in the network can be easily identified. In addition, the data stored in IPFS also has the same property, if you want to modify the data, it will affect the hash value in the blockchain network. If the value of one of the participants is modified, the remaining nodes will not be affected by this. Therefore, any kind of modification will be recognized immediately.

7. Experimental evaluation

In this section, the functions of this scheme are compared with those of other EMRs schemes, and then the performance of the scheme is evaluated by numerical simulation experiments.

7.1. Experimental setup

The operating system of this experiment simulation environment is Windows11, the processor is Intel(R) Core(TM) i7-10750H CPU @ 2.60 GHz 2.59 GHz, and the memory is 16 GB. Java is used to realize the scheme in this paper, and the JPBC library is used. The base field of the elliptic curve used is 512 bit and the curve is Type-A [35]. Since EMRs datasets involve personal privacy, this section uses the UC Irvine machine learning database, in which NSF Research Award Abstracts 1990–2003 data sets are selected as alternatives [36].

7.2. Functional analysis

The performance comparison of BMSE with the other three schemes is given in Table 2. Literature [9] only provides privacy protection features without applying blockchain, multi-keyword, and authentication, literature [12] implements privacy protection and authentication but does not apply blockchain and use multi-keyword, literature [13] only does not use multi-keyword but supports the other three features, and literature [33] uses K-means to cluster documents, similar to this scheme, but not applied to blockchain. BMSE makes up for the shortcomings of these related work, and applies the solution to the blockchain, adopting the multi-keyword search method and also conducting identity verification during the search process, which well protects the privacy of users.

Table 3
Computation time of common cryptographic algorithms (ms).

Operation	T_p	T_e	T_m	T_h
Time	5.655	2.063	0.017	0.009

Table 4
Comparison of calculation time of each stage (ms).

Scheme	Encryption time	Search time
[13]	$T_p + 4T_e + 7T_m = 14.03$	$T_p + 5T_e + 3T_m + 2T_h = 16.04$
[33]	$2T_p + 3T_e + 2T_h = 17.52$	$2T_p + T_e + 3T_m + T_h = 13.43$
BMSE	$T_p + 3T_e + 5T_m = 11.93$	$T_p + 3T_h = 5.68$

7.3. Computation cost analysis

In Table 3, T_p represents the time of the bilinear pairing operation, T_e represents the time of the exponential operation, T_m represents the time of the multiplication operation, and T_h represents the time of the hash operation. The order of pairing time of common cryptographic operations is $T_p > T_e > T_m > T_h$, and the time T_p of bilinear pairing operation is much larger than the time of other cryptographic operations.

In Table 4, we calculate the time of literature [13], literature [33] and BMSE scheme in the search phase and the decryption phase.

As can be seen from the table, when encrypting data, the calculation amount of each scheme from large to small is in the order of literature [33], literature [13], and BMSE. In the data search, the calculation amount of each scheme from large to small is in the order of literature [13], literature [33], and BMSE.

In summary, the BMSE scheme is superior to the other schemes from the results of the functional analysis and the comparison of the computation time of each stage.

7.4. Numerical analysis

This section evaluates the effectiveness of BMSE in terms of search accuracy, search time, and cost analysis.

7.4.1. Clustering time analysis

Literature [33] requires the direct clustering of documents, but because direct clustering takes longer when there are many documents, we think about computing the Simhash value of each document before clustering it, which can speed up the process considerably. The findings of our experimental evaluation of both direct clustering and computing Simhash values prior to clustering are displayed in Fig. 3. As can be seen from the figure, when the number of documents is 1000, 1500, 2000, the time consumed by directly clustering documents and the time consumed by clustering after Simhash is calculated is almost the same. However, as the number of documents gradually increases to 2500, 3000, 3500, The time to cluster documents directly is significantly longer than the time to cluster after calculating Simhash value first. This shows that when the number of documents is large, the documents need to be processed first, and the Simhash value of the documents is used as the identification of the documents for clustering, which can significantly improve efficiency.

Table 5
Search accuracy.

l	1000	1500	2000	2500	3000	3500
Δp	0.828	0.837	0.830	0.835	0.837	0.830

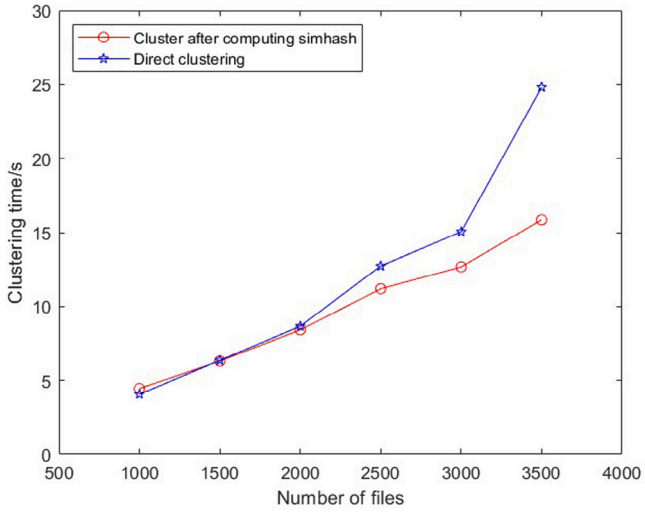


Fig. 3. Clustering time.

7.4.2. Search accuracy

When CSP returns the top k documents with the highest similarity scores based on indexes, some documents with high similarity scores are missed because they are searched in clusters. To evaluate the efficiency of our search, the metric of precision is adopted to evaluate our method. The evaluation effects are shown in the Table 5. We use the precision rate calculation formula defined in Ref. [37] to calculate the precision rate. The calculation formula is shown in formula (6):

$$\Delta p = \frac{l'}{l} \tag{6}$$

Where l' is the true number of top-k documents returned by CSP, and l is the number of documents that should be returned. From Table 5, it can be seen that when the number of returned documents is 1000, Δp is 0.828; when the number of returned documents is 3500, Δp is 0.830. It can be seen that Δp is almost unaffected by the number of returned documents, and as the number of returned documents gradually increases, the accuracy rate can remain stable at about 0.83.

7.4.3. Search time evaluation

For both scenarios of utilizing and not using the clustering algorithm for document sets, the search time of the scheme in this work is compared. The search time is evaluated and examined in terms of the quantity of documents and keywords searched, respectively.

The time required for an authorized user to search a document by the number of keywords is shown in Fig. 4. If we do not use the clustering operation on the document, when the number of search keywords is 10, the search time needs 300 ms, when the number of search keywords is 50, the search time needs 812 ms. When the clustering operation is used, the search time is significantly improved. When the number of search keywords is 10, the search time only needs 120 ms, and when the number of search keywords is 50, the search time only needs 348 ms.

In Fig. 5, we measured the search time prior to and following clustering for totals of 1000, 1500, 2000, 2500, 3000, and 3500 documents, respectively. As can be seen from the figure, when the clustering operation is not used, the search time is 152 ms when the total number of documents is 1000, and 434 ms when the total number of documents is 3500. When the clustering operation is used, the search time is only

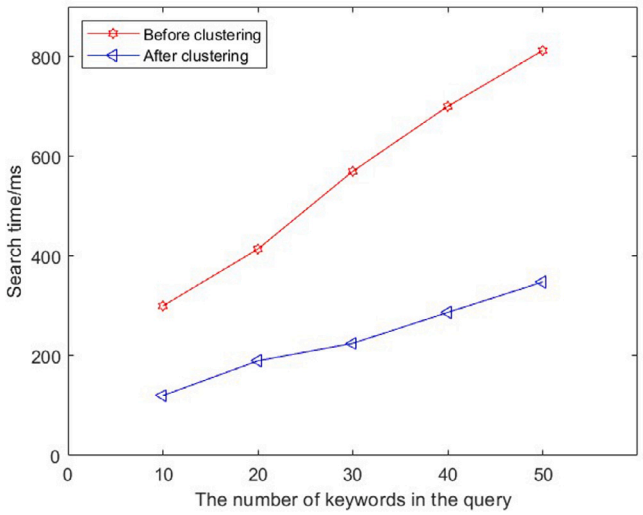


Fig. 4. Keyword-based search time.

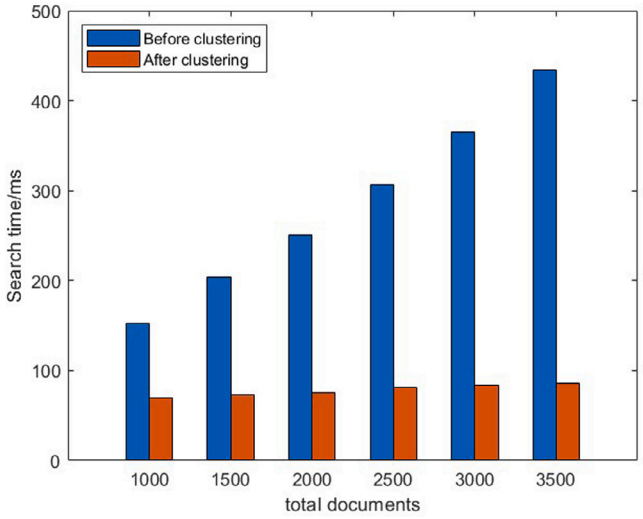


Fig. 5. Search time based on total documents.

70 ms when the total number of documents is 1000, and 85.7 ms when the total number of documents is 3500. Therefore, the search efficiency with clustering is much higher than that without clustering, and it can also be seen from the table that the search time with clustering is almost unaffected by the total number of documents.

Fig. 6 shows that the search time increases with the number of documents returned, As can be seen from the figure, before the clustering operation is used, when the number of returned documents is 1000, it takes 300 ms, and when the number of returned documents is 3500, it takes 950 ms. However, after clustering, when the number of documents returned is 1000, it only takes 120 ms, and when the number of documents returned is 3500, it only takes 210 ms. It can be seen that in terms of the total number of documents returned, the search time can still be significantly reduced after using clustering.

The reasons for the reduction of BMSE search time based on the number of search keywords, total number of documents, and number of returned documents are as follows:

- In the file processing algorithm, we use K-means algorithm for the document set, and divide the document set into k clusters, and the files in each cluster are similar

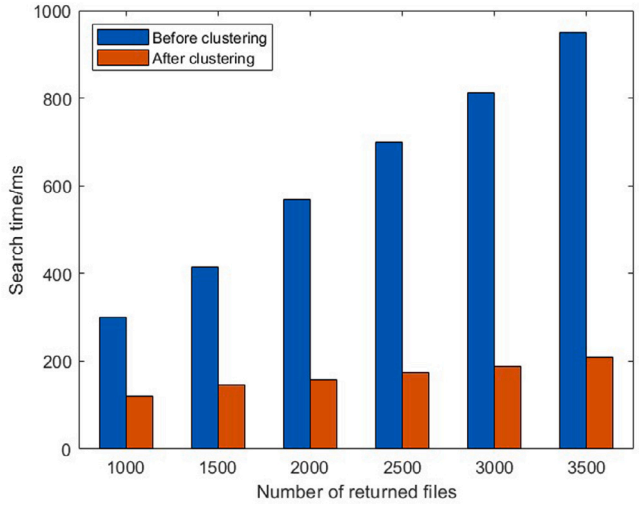


Fig. 6. Search time based on return documents.

Table 6
Smart contract evaluation.

Function	GAS	GWEI	ETH
Contract deployment	996 134	2.3	0.00229111
Get index	48 315	2.3	0.00011112
Get result set	60 158	2.3	0.00013836

- When we search, we search in each well-clustered cluster, rather than searching the entire document set each time, reducing the number of comparisons during the search and therefore reducing the search time

7.4.4. Smart contract evaluation

We evaluated the overhead on smart contracts based on the online Remix IDE, and the results are shown in Table 6: The experiment measures consumption in GAS, and when running, the code uses a specific amount of GAS. Each transaction's priority price is its GWEI, and its cost is the sum of its GAS and GWEI. 996134 gases at a cost of 0.00229111 are used, together with 48315 gases at a cost of 0.00011112 for obtaining the index and 60158 gases at a cost of 0.00013836 for obtaining the query result set, in order to effectively launch the smart contract. The cost of getting indexes and result sets is quite low, and the deployment contract consumes the most money, as shown in the table. This indicates that the method proposed in this work can implement the search function with little overhead.

8. Conclusion

This paper proposes an EMRs multi-keyword searchable encryption scheme based on blockchain. Because BMSE is applied to blockchain, it can effectively solve the problem that centralized CSP can lead to the theft of patient privacy. At the same time, BMSE can reduce the number of comparisons between keyword indexes during the search process. In addition, the multi-keyword retrieval of encrypted data reduces the retrieval cost and improves the retrieval efficiency. Finally, the data and the keyword index are encrypted using AES and ABSE respectively to ensure their security. The experiment shows that the scheme is effective. In the future work, we will continue to study the possible problems of encryption on the blockchain and the latest progress of cryptography, so as to be able to design more efficient and secure searchable encryption schemes and put them into the practical environment for application.

CRediT authorship contribution statement

Shen Fanfan: Conceptualization, Methodology. **Shi Lin:** Data curation, Investigation, Writing – original draft. **Zhang Jun:** Supervision. **Xu Chao:** Supervision. **Chen Yong:** Project administration. **He Yanxiang:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

[1] Shufen Niu, Wenke Liu, Lixia Chen, Caifen Wang, Xiaoni Du, A searchable encrypted electronic medical record data sharing scheme based on consortium blockchain, J. Commun. 41 (204–214) (2020).
[2] Guipeng Zhang, Zhenguo Yang, Wenyin Liu, Blockchain-based privacy preserving e-health system for healthcare data in cloud, Comput. Netw. 203 (2022) 108586.
[3] Lubin Lin, Guipeng Zhang, Yanfeng Li, Zhenguo Yang, Wenying Liu, A searchable encrypted scheme for blockchain-based healthcare systems, J. Mini-Micro Syst.
[4] Christian Esposito, Alfredo De Santis, Genny Tortora, Henry Chang, Kim-Kwang Raymond Choo, Blockchain: A panacea for healthcare cloud-based data security and privacy? IEEE Cloud Comput. 5 (1) (2018) 31–37.
[5] Jingzhong Wang, Mengru Li, Yunhua He, Hong Li, Ke Xiao, Chao Wang, A blockchain based privacy-preserving incentive mechanism in crowdsensing applications, IEEE Access 6 (2018) 17545–17556.
[6] Zheng Yao Ng, Iftekhar Salam, Blockchain-based multi-keyword search on encrypted COVID-19 contact tracing data, in: International Conference on Information Security Practice and Experience, Springer, 2022, pp. 75–92.
[7] Paul Tak Shing Liu, Medical record system using blockchain, big data and tokenization, in: Information and Communications Security: 18th International Conference, ICICS 2016, Singapore, Singapore, November 29–December 2, 2016, Proceedings 18, Springer, 2016, pp. 254–261.
[8] Dawn Xiaoding Song, David Wagner, Adrian Perrig, Practical techniques for searches on encrypted data, in: Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000, IEEE, 2000, pp. 44–55.
[9] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, Public key encryption with keyword search, in: Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings 23, Springer, 2004, pp. 506–522.
[10] Amit Sahai, Brent Waters, Fuzzy identity-based encryption, in: Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings 24, Springer, 2005, pp. 457–473.
[11] Wenhai Sun, Shucheng Yu, Wenjing Lou, Y Thomas Hou, Hui Li, Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud, in: IEEE INFOCOM 2014-IEEE Conference on Computer Communications, IEEE, 2014, pp. 226–234.
[12] Xiao Wang, Aiqing Zhang, Xiaojuan Xie, Xinrong Ye, Secure-aware and privacy-preserving electronic health record searching in cloud environment, Int. J. Commun. Syst. 32 (8) (2019) e3925.
[13] Jingwei Liu, Xiaolu Li, Lin Ye, Hongli Zhang, Xiaojiang Du, Mohsen Guizani, BPDS: A blockchain based privacy-preserving data sharing for electronic medical records, in: 2018 IEEE Global Communications Conference, GLOBECOM, IEEE, 2018, pp. 1–6.
[14] Shahzaib Tahir, Hasan Tahir, Ali Sajjad, Muttukrishnan Rajarajan, Fawad Khan, Privacy-preserving COVID-19 contact tracing using blockchain, J. Commun. Netw. 23 (5) (2021) 360–373.
[15] Mehran Mozaffari-Kermani, Reliable and High-Performance Hardware Architectures for the Advanced Encryption Standard/Galois Counter Mode (Ph.D. thesis), The University of Western Ontario (Canada), 2011.
[16] Mehran Mozaffari Kermani, Reza Azarderakhsh, Integrating emerging cryptographic engineering research and security education, Am. Soc. Eng. Educ. (ASEE) (2015).
[17] Ausmita Sarker, Mehran Mozaffari Kermani, Reza Azarderakhsh, Efficient error detection architectures for postquantum signature falcon's sampler and KEM SABER, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 30 (6) (2022) 794–802.

[18] Ausmita Sarker, Alvaro Cintas Canto, Mehran Mozaffari Kermani, Reza Azarderakhsh, Error detection architectures for hardware/software co-design approaches of number-theoretic transform, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. (2022).

[19] Mehran Mozaffari Kermani, Reza Azarderakhsh, Jiafeng Xie, Error detection reliable architectures of camellia block cipher applicable to different variants of its substitution boxes, in: 2016 IEEE Asian Hardware-Oriented Security and Trust, AsianHOST, IEEE, 2016, pp. 1–6.

[20] Mehran Mozaffari Kermani, Erkay Savas, Shambhu J. Upadhyaya, Guest editorial: Introduction to the special issue on emerging security trends for deeply-embedded computing systems, IEEE Trans. Emerg. Top. Comput. 4 (3) (2016) 318–320.

[21] Alvaro Cintas-Canto, Jasmin Kaur, Mehran Mozaffari-Kermani, Reza Azarderakhsh, ChatGPT vs. Lightweight Security: First Work Implementing the NIST Cryptographic Standard ASCON, 2023, arXiv preprint arXiv:2306.08178.

[22] Reza Azarderakhsh, Brian Koziel, SH Fatemi Langroudi, Mehran Mozaffari Kermani, FPGA-SIDH: High-performance implementation of supersingular isogeny Diffie-Hellman key-exchange protocol on FPGA, 672, 2016, pp. 1–18, Proc. eprint 2016.

[23] Elena Dubrova, Kalle Ngo, Joel Gärtner, Ruize Wang, Breaking a fifth-order masked implementation of crystals-kyber by copy-paste, in: Proceedings of the 10th ACM Asia Public-Key Cryptography Workshop, 2023, pp. 10–20.

[24] Jasmin Kaur, Alvaro Cintas Canto, Mehran Mozaffari Kermani, Reza Azarderakhsh, A comprehensive survey on the implementations, attacks, and countermeasures of the current NIST lightweight cryptography standard, 2023, arXiv preprint arXiv:2304.06222.

[25] Alvaro Cintas Canto, Jasmin Kaur, Mehran Mozaffari Kermani, Reza Azarderakhsh, Algorithmic security is insufficient: A comprehensive survey on implementation attacks haunting post-quantum security, 2023, TechRxiv.

[26] Xueqiao Liu, Guomin Yang, Yi Mu, Robert H. Deng, Multi-user verifiable searchable symmetric encryption for cloud storage, IEEE Trans. Dependable Secure Comput. 17 (6) (2018) 1322–1332.

[27] C. Rama Krishna, Rohit Handa, et al., Dynamic cluster based privacy-preserving multi-keyword search over encrypted cloud data, in: 2016 6th International Conference-Cloud System and Big Data Engineering, Confluence, IEEE, 2016, pp. 146–151.

[28] C. Rama Krishna, Rohit Handa, et al., Dynamic cluster based privacy-preserving multi-keyword search over encrypted cloud data, in: 2016 6th International Conference-Cloud System and Big Data Engineering, Confluence, IEEE, 2016, pp. 146–151.

[29] Jiangang Shu, Kan Yang, Xiaohua Jia, Ximeng Liu, Cong Wang, Robert H Deng, Proxy-free privacy-preserving task matching with efficient revocation in crowdsourcing, IEEE Trans. Dependable Secure Comput. 18 (1) (2018) 117–130.

[30] Weiran Liu, Xiao Liu, Qianhong Wu, Bo Qin, Experimental performance comparisons between (h) ibe schemes over composite-order and prime-order bilinear groups, in: Proceedings of 2014 11th International Bhurban Conference on Applied Sciences & Technology, IBCAST Islamabad, Pakistan, 14th-18th January, 2014, IEEE, 2014, pp. 203–209.

[31] Ahmed M. Manasrah, Mahmoud Abu Nasir, Maher Salem, A privacy-preserving multi-keyword search approach in cloud computing, Soft Comput. 24 (8) (2020) 5609–5631.

[32] Syam Kumar Pasupuleti, Subramanian Ramalingam, Rajkumar Buyya, An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing, J. Netw. Comput. Appl. 64 (2016) 12–22.

[33] Z.J. Fang, Shu Zhou, Z.H. Xia, Research on fuzzy search over encrypted cloud data based on keywords, Comput. Sci. 42 (3) (2015) 136–139.

[34] David A. McGrew, John Viega, The security and performance of the galois/counter mode of operation (full version), 2004, Cryptology ePrint Archive.

[35] Angelo De Caro, Vincenzo Iovino, jPBC: Java pairing based cryptography, in: 2011 IEEE Symposium on Computers and Communications, ISCC, IEEE, 2011, pp. 850–855.

[36] K. Bache, M. Lichman, NSF research award abstracts 1990–2003 data set, UCI Mach. Learn. Repository (2013).

[37] Zhu Xiangyang, Dai Hua, Yi Xun, Yang Geng, Li Xiao, et al., MUSE: an efficient and accurate verifiable privacy-preserving multikeyword text search over encrypted cloud data, Secur. Commun. Netw. 2017 (2017).