SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Privacy-Preserving Phrase Search Algorithm Using Bloom Filters and Hybrid AES_GCM for Secured IoT Data in Cloud Environments

Surakanti Sphoorthy Reddy[1], Dr. Ramachandram Sirandas[2]

[1] Ph.D. Scholar, Anurag University, Hyderabad, India. E-mail: sphoorthy.surakanti@gmail.com, ORCID ID: 0009-0005-8673-3540
[2] Professor, Department of Computer Science and Engineering, Osmania University, Hyderabad, India
E-mail: schandram@gmail.com, ORCID: 0000-0002-9156-2302

Abstract
The growth of IoT devices creates huge amounts of sensitive data that are usually sent to cloud platforms for storage and processing. But making it possible to do a secure and quick phrase search on encrypted IoT data is still hard because of privacy concerns and limited computing power. This paper suggests a new phrase search algorithm that protects privacy by combining Bloom Filters and a hybrid AES_GCM encryption scheme. Space-efficient, probabilistic search indexes are made with Bloom Filters that can be used to match multiple keywords and phrases with minimal additional storage space and false-positive rates. The AESGCM hybrid module secures the privacy and integrity of data by authenticated encryption. This ensures that the plaintext is less likely to be leaked or ciphertext will be altered when storing or transmitting the data in the cloud. Strict performance test and evaluation indicate that the suggested algorithm improves search accuracy, query response time, and computation and efficiency significantly than the conventional encrypted search techniques. Also, the scheme is very hard to break with inference and frequency-based attacks, which makes IoT-generated datasets even more private from end to end. The framework can handle dynamic updates and works well with different types of IoT data streams and queries of different levels of difficulty. The work addresses essential security gaps in data searchability outsourcing and provides an easy and convenient method of obtaining data as well as guarding privacy in IoT-cloud environment with limited resources. It also preconditions the further safe IoT data handling designs.

Keywords- Bloom Filters, hybrid AES_GCM, false-positive, ciphertext and query response time.

1. Introduction

The Internet of Things (IoT) is changing quickly, and smart devices and sensors that are connected to each other in different fields, like healthcare, smart homes, industrial automation, and smart cities, create huge amounts of data (Chen et al., 2022). Cloud computing has turned out to be a necessary option in storing, managing and processing this large volume of data due to its ability to provide scalable storage and strong computing capabilities. Nonetheless, transmitting sensitive IoT data to the cloud poses a big question in regard to privacy, security and unauthorized access.

As a result, making sure that data is stored safely while also making it easy to find, especially through phrase-based search, has become a major research problem.

In the current IoT applications, users tend to phrase search to get information that is more helpful to them. Search using traditional methods of keywords does not work well in this. It takes new ideas to be able to run such complicated queries on encrypted data stored in the cloud without giving up privacy. A good direction to take is to employ secure searchable encryption schemes (Yan et al., 2023). These search through encrypted data without letting untrusted cloud servers see what it is.

Bloom Filters are space-efficient probabilistic data structures that quickly and easily match phrases while using as little storage and processing power as possible (Chen et al., 2021; Liu & Chen, 2020; Wang et al., 2021; Singh et al., 2022). This is very important for IoT environments with limited resources. The hybrid AES_GCM encryption protects IoT data by using authenticated encryption to keep it private and safe from unauthorized access and tampering while it is being stored and sent.

In order to address these issues, this work proposes a new Privacy-Preserving Phrase Search Algorithm that can be used to solve the problems by combining the high bandwidth of Bloom Filters with the high security of a hybrid Advanced Encryption Standard-Galois/Counter Mode (AESGCM). By having a combination of Bloom Filters and AESGCM, the proposed method offers a good balance between security, efficiency and usability. Phrases may be searched in encrypted data by users without providing the cloud provider with access to confidential information, which preserves end-to-end privacy. This framework has

a lot of potential for different IoT situations. It can assist cloud-assisted IoT systems to manage information in a safer manner and assist in making an IoT-based ecosystem trustworthy and respecting privacy.

1.1 Contribution Statement

This research proposes a new privacy-saving architecture of the security of phrase search with encrypted IoT information in a cloud-based setup. Its key contributions include: (1) a hybrid design that incorporates Bloom Filter-based phrase indexing with AES_GCM authenticated encryption to provide efficient search and high data privacy; (2) a resource-efficient phrase search algorithm that fits well in resource-constrained IoT devices; (3) an end-to-end secure workflow that does not allow cloud servers to access plaintext data or search content; (4) better search accuracy by phrase-level matching than traditional key-word based encrypted search solutions; and (5) a scalable design that can be utilized in large- The suggested solution improves the security, efficiency, and usability ratio in IoT privacy sensitive data management systems.

2. Related Works

In the context of the Internet of Things (IoT), data dynamics, which include adding, deleting, and updating data often, make things even more complicated. Chen et al. (2021) came up with a solution by making an encrypted index structure for phrase search that can be updated easily. Their plan combined dynamic SSE with lightweight Bloom Filters, which gave them a moderate query performance. Still, it cost a lot of money to re-encrypt indexes every time they were updated.

Hybrid authenticated encryption schemes have become more popular as a way to protect the security of the data they are based on. Recent research indicates that the application of AES in conventional mode such as CBC that lacks an inbuilt integrity check is not a good idea. Galois/Counter Mode (AESGCM) has become rather a powerful alternative that offers the privacy and integrity protection. Kumar and Das (2021) tested AES_GCM in IoT settings and found that it effectively stops tampering attacks during cloud storage and transmission without adding too much computational overhead, which is important for IoT devices that don't have a lot of resources.

Liu and Chen (2020) came up with a better way to search for phrases that uses a dual index mechanism. This means that a secure positional index is placed on top of a keyword index to keep the order of the words. Park et al. (2022) also showed that AES_GCM can be used in edge-cloud collaborative architectures in real life. They showed that it has low latency and is resistant to replay and modification attacks. Even so, combining AES_GCM with search indexes that are both fast and protect privacy is still not very well understood. There have only been a few attempts to combine it with phrase search capabilities.

New methods Rahman et al. (2022) have also focused on edge-assisted secure search to take some of the work off of the cloud. But making sure that edge and cloud are securely encrypted and synced without any problems is still a research problem.

Singh et al. (2022) went a step further and suggested a hierarchical Bloom Filter structure that includes positional encodings. This hybrid index search for phrases without taking up too much space, but their model still showed some access pattern leaks that attackers might be able to use.

Another important change in the last four years has been the use of probabilistic data structures to deal with problems with index size. Wang et al. (2021) looked into Bloom Filter-based secure search frameworks for situations where one need to search for more than one keyword or phrase. However, most of the existing implementations were only able to handle multi-keyword sets and didn't do full phrase order matching, which is very important for IoT applications that need to be sensitive to context, like healthcare monitoring and smart surveillance.

Researchers have also looked at ways to reduce privacy leaks, which is something that has been left out of many earlier studies. Xu et al. (2023) came up with an SSE scheme that is resistant to inference attacks by making trapdoor generation random and hiding query patterns.

Ali et al. (2023) came up with a phrase search algorithm that takes privacy into account by using compressed Bloom Filters and secure hash functions. Their framework did a good job of reducing the amount of communication that had to happen between IoT edge devices and the cloud. However, it had some problems when it was used on a large scale, when the false positive rate of Bloom Filters became harder to control.

Recent researches have concentrated on improving searchable symmetric encryption (SSE) schemes. Zhang et al. (2021) developed an SSE model that facilitates phrase queries through the combined use of inverted indexes and position vectors.

Reference scheme

This is a follow up and extension to the Bloom-filter based searchable encryption methodology proposed in earlier literature (see the base reference(s) in the original submission). The baseline scheme to be used as a reference during algorithmic design and comparison is the scheme of "Bloom-filter-based SSE" in which per-document Bloom filters represent key-membership and trapdoors are tokens generated using HMAC to be used in membership-testing on the server. The main particulars of our scheme are:

We encourage phrase-level search (as opposed to single-keyword membership) by indexing ordered n-grams and by adding trapdoor construction which maintains a phrase ordering operation in the matching. We give verbatim Bloom filter sizing calculations along with sensitivity analysis and experimentally give the impact of false-positive as well as the run-time overhead. Table 1 compares existing searchable encryption and Bloom Filter-based phrase search schemes with the proposed framework.

Table 1. Comparison chart for proposed results in comparison with state-of-the-art studies

Ref	Index Structure	Encryption	Phrase Support	Trapdoor Generation Time	Search Time	Storage Overhead	Security Level	False Positive Control
Yan et al. (2023)	Dynamic Bloom Filter	AES-CBC	Yes	Moderate	Moderate	Medium	No integrity protection	Moderate
Gao et al. (2023)	Positional Index	AES	Yes	High	Fast	High	Vulnerable to inference	Low
Yang et al. (2020)	Hierarchical Bloom Filter	AES	Yes	Moderate	Moderate	Medium	Partial leakage risk	Moderate
Chen et al. (2021)	Compressed Bloom Filter	Lightweight AES	Yes	Fast	Moderate	Low	Limited authentication	High (large scale issue)
Liu and Chen (2020)	Secure SSE Index	AES	Partial	High	Moderate	High	Strong privacy	Very Low
Rahman et al. (2022)	Bloom Filter	AES	Multi-keyword	Moderate	Moderate	Low	Moderate	Moderate
Proposed Method	Encrypted Bloom Filter	AES-GCM Authenticated Encryption	**Full phrase support**	**Very Fast (50% faster)**	**Fast**	**Low**	**Very High (Confidentiality + Integrity + Authentication)**	**Optimized and Controlled**

2.1 Research Gap and Novelty of the Proposed Scheme

Although there have been studies on SSE using Bloom Filter, Phrase Search and AES-GCM-based secure storage, there are still some important questions that have not been addressed. Bloom Filter SSE schemes currently only support single-keyword membership testing and cannot efficiently maintain the consecutive relationship between keywords such as phrase matching (Peng et al., 2024). Positional indexes typically use extra storage structures that cause extra costs in the computation and communication (Shiraly et al., 2024). Moreover, a large number of phrase-search frameworks only consider the search aspect while neglecting the inclusion of authenticated encryption technologies that can ensure the confidentiality, integrity and authenticity of the outsourced IoT data (Peng et al., 2024). One of the drawbacks is that many solutions currently available apply encryption and searchable indexing as distinct modules (Zhou et al., 2024). This means that phrase verification needs to involve several traversals of the indexes, and/or comparisons of vectors at position, and/or computation on the server side (Zhang et al., 2021). These are not well suited in resource-limited IoT deployments where storage efficiency and low latency to queries are paramount. The proposed scheme overcomes these limitations by combining three mutually complementary mechanisms in a single scheme. First, the n-grams phrases are ordered before processing and then they are directly encoded into the Bloom Filters. This allows for Phrase Matching as well as Ordering of Keywords. Second, a new generation mechanism is added to generate phrase-aware search tokens instead of keywords. Therefore, it is possible to check the membership of a phrase using only the Bloom Filter and without any additional positional indexes. Third, AES-GCM authenticated encryption has been embedded in the searchable indexing framework to ensure both confidentiality and integrity of outsourced IoT data at the same time. The proposed method is similar to the conventional Bloom Filter SSE model in which the document Bloom Filter stores the tokens of the keywords in the document, except that it stores ordered phrase tokens. A phrase query trapdoor is generated for the ordered phrase segments for a phrase query. A document is only a match if all of the ordered phrase trapdoors are successfully verified

in the Bloom Filter. This phrase-preserving indexing technique helps to minimize ambiguity in the independent keyword matching process and enhances the precision of the phrase search. So, the novelty of the proposed work is not in the individual use of Bloom Filters, SSE or AES-GCM only but in the integration of all three: (i) the ordered indexing of n-grams with Bloom Filters, (ii) the construction of trapdoors based on the phrases, and (iii) the secure and efficient retrieval of phrases in IoT cloud setups protected by AES-GCM. To the best of our knowledge, few existing studies combine these three components into a single privacy-preserving phrase search framework specifically optimized for lightweight IoT applications. Overall summary on comparison with existing techniques is given in Table 2.

Table 2. Comparison of Existing Approaches and Proposed Scheme

Author	Search Type	Security Level	Key Drawback
Liu and Chen (2020) [9]	Phrase Search	Moderate	Large positional index and increased storage overhead
Chen et al. (2021) [7]	Dynamic SSE	High	Frequent re-encryption during updates increases computational cost
Wang et al. (2021) [13]	Bloom Filter Search	Moderate	Limited phrase-order matching capability
Singh et al. (2022) [12]	Hierarchical Bloom Filter	Good	Access-pattern leakage remains possible
Ali et al. (2023) [15]	Privacy-Preserving Phrase Search	High	Bloom Filter false positives increase at large scale
Xu et al. (2023) [14]	Inference-Resistant SSE	High	Additional complexity in trapdoor generation
Proposed Scheme	**Phrase Search-AES-GCM**	**High**	**Addresses the above limitations through compact indexing, phrase-aware trapdoors, and authenticated encryption**

The comparison made is in relation to representative schemes reported by the literature recently. Some of these are Bloom Filter based SSE schemes, introduced by Wang et al. (2021) and Zhang et al. (2021), positional-index phrase search techniques introduced by Liu and Chen (2020) and Zhang et al. (2021) as well as AES-GCM based secure storage schemes introduced by Kumar and Das (2021) and Park et al. (2022). Our scheme enhances these techniques to build a unified private phrase search system with ordered n-gram Bloom Filter indexing, phrase-aware trapdoor generation and AES-GCM authenticated encryption.

3. Research Methodology

3.1 Dataset Description

Dataset Name

Medical Abstracts – Text Classification Corpus

Link: https://github.com/sebischair/medical-abstracts-tc-corpus

Dataset Overview

-
- The medical abstracts in the dataset include information about the health conditions of patients.
- It is intended to be used on text classification and information search duties.
- Typically used to create simulation of real-world healthcare data to search with the objective of privacy preservation.

Table 3. Key Dataset Details

Feature	Description
Total Samples	~11,550 training samples
Classes	5 different classes (disease categories)
Data Format	Text abstracts with condition descriptions
Language	English
Type	Open-source, public medical text dataset

Classes in Dataset

Each abstract belongs to one of these 5 classes:

- Digestive System Neoplasms

- Nervous System Disorders
- Cardiovascular Conditions
- General Conditions
- Other Miscellaneous Conditions

The dataset in question is a set of medical abstracts that were gathered among patients, and the abstracts were divided into five different classes of medical conditions. The classes are a representation of different health conditions and diagnostic groups allowing the dataset to be used as a good reference point to check phrase search algorithms in sensitive healthcare settings. The training dataset (11550 samples) is adequate to check the efficiency and strength of privacy-preserving search methods.

In the case of cloud-assisted IoT healthcare, the medical abstracts of patients that have been encrypted should still be searchable to allow clinicians and other authorized researchers to retrieve the necessary data. However, both security and utility are hard to ensure, especially when dealing with strictly personal identifiable and highly sensitive data on medical information. Through its combination with Bloom filters, the text abstracts of the dataset can be coded into privacy-enabling index data format, which allows to query the dataset in terms of phrases without disclosing raw medical data. Moreover, a hybrid of AES-GCM encryption scheme will provide data confidentiality and integrity when storing and retrieving data in the cloud.

The medical abstracts dataset is not only an alternative source of various textual data to assess the accuracy of search, false positives, and query performance but also a mirror of issues in the real world medical systems that are based on IoT in which scalability, latency, and secure accessibility need to be balanced. In such a way, the given data set constitutes a crucial basis to justify the efficiency of the suggested privacy-conserving phrase search model.

3.2 Data Pre-processing

Textual medical data preprocessing is an important procedure in the design of efficient and secure search algorithms in IoT-enabled cloud systems. In this regard, removal of stopwords on the medical abstracts is one of the most important operations done on the dataset. Stopwords are frequently occurring words like the, is, and, and, which do not have much semantic importance to the analysis of medical phrases. With such a system, the occupations that are redundant are removed and the system is left only with the important, meaningful and alphabetic tokens, which are crucial in indexing and precision of search.

The operationalization of this will consist of using a function called removestopwords that will be used systematically throughout each row of the medicalabstract column in the dftrain DataFrame. Every medical abstract is undergone to remove irrelevant stopwords so that only medically relevant terms and phrases are left. The optimized and cleaned copy of every abstract will then be stored under a new column called cleanedabstract. It is this extra column that is the foundation of building privacy-compliant Bloom filter indices that enable the safe phrase search of encrypted data.

The Bloom filters of every record are encrypted with the AES-GCM and saved.

- A trapdoor is generated by the user when he/she types a phrase and invokes a hash and then encrypts the query to the same Bloom-filter format.
- In search, encrypted indices of all the stored indexes are unencrypted in a temporary manner to make comparisons.
- The trapdoor and decrypted index are both changed into bitarrays.
- These bitarrays are compared and the percentage of matching bits is calculated by the system.
- The ranking of records is conducted on the basis of this matching score or relevancy of the abstract to the search phrase..

Some of the benefits of eliminating the stopwords are that not only do they minimize the overhead of storage and indexing, but also the likelihood of finding false matches in the encrypted search operations is also minimized. Combined with the AES-GCM Authenticated Encryption mechanism, the system will ensure confidentiality and verifiability of abstracts of patient data stored. Also, the privacy-preserving phrase search algorithm is more scalable when the text is optimally preprocessed, especially in the contexts where the medical data streamed by IoT devices constantly flows into the clouds.

3.3 Phrase search specifics

Definition. A phrase is a sequence of words that are consecutive in the text of the document.

Maximum phrase length. In this work we took a very modest upper limit w_{max} = 5 words (parameter). The system allows w 1 to wmax, where a query longer than wmax is divided into wmax-size overlapping windows to match.

Matching semantics. Phrase matching is contiguous: the match of a a phrase p = (t1, t2, etc, tw) may only be ordered; in particular, the token sequence must appear in a document in a continuous manner. In the case of longer queries (length L is greater than wmax) the server only reports a match when all smaller wmax windows are matched (logical AND between windows) generating some approximation of long query semantics.

Test Case: When w max is set to 5, query of the type secure Bloom filter membership (3 words) is indexed and searched as a 3-gram. A query consisting of 9 words is divided into five overlapping 5-grams; all of them have to be present to make the document be returned as a complete match.

AES_GCM based Encryption Framework

The AESGCM (Advanced Encryption Standard with Galois/Counter Mode) is a powerful and effective symmetric encryption protocol that is common in modern applications to protect the privacy of information, such as cloud computing and the Internet of Things (IoT). This makes the data of both parties private and secure. AES block cipher is used in the counter mode (CTR) in order to convert plaintext into ciphertext blocks. At the same time, it uses Galois field multiplication process to create an authentication tag that ensures integrity of the data when transferred or stored.

AESGCM has the benefit of associating data and encryption (AEAD) being supported. This implies that it is possible to verify other unencrypted metadata including headers alongside the ciphertext. This is an especially useful feature of secure IoT structures and network communications. AESGCM is quite fast and can be executed parallel which makes its latency minimal and its throughput remarkable. This is best fitted in devices with limited resources and applications that need real time performance. There are standards that NIST has defined regarding its usage and many cryptographic libraries implement it. AESGCM is a useful tool because it satisfies the needs of a strong encryption process and a trustworthy authentication. This improves the safety of information in different digital systems.

3.4 Bloom Filter Construction

A Bloom Filter is very important for IoT data cloud storage. It quickly and accurately match keywords or phrases without giving away personal information. A Bloom Filter is a probabilistic data structure which allows usersto query whether a specific piece of data belongs or not in a manner which occupies minimal space, and has neither false positives nor false negatives.

A Bloom Filter is represented as a bit vector of length m as shown in Eq. (1)

$$BF = \{b_1, b_2, \dots, b_m\} \quad (1)$$

$$b_i \in \{0,1\}, 1 \le i \le m \quad (2)$$

Here, the term m denotes the size of the Bloom Filter.

Let $P = \{p_1, p_2, \cdots, p_n\}$ represent the set of phrases extracted from a document. Each phrase is mapped using k independent hash functions as in Eq. (2).

$$h_j : P \to \{0,1, \dots, m-1\}, j = 1,2, \dots, k \quad (3)$$

For each phrase p_i, the hash functions generate k positions within the Bloom Filter. The insertion operation is defined as in Eq. (4).

$$BF[h_j(p_i)] = 1, \forall j = 1,2, \dots, k \quad (4)$$

The probability of false positives in the Bloom Filter is given as in Eq. (5).

$$P_{fp} = \left(1 - e^{-kn/m}\right)^k \quad (5)$$

where:

- m= Bloom filter size
- k= number of hash functions
- n= number of inserted phrases

This equation is used to determine appropriate Bloom Filter parameters that balance storage efficiency and search accuracy. The individual that owns the data initially takes the text and divides it into keywords to construct it. A hash is done on each keyword using multiple independent hash functions and each key is mapped to a few positions in a fixed sized bit array.

Set the bits at these positions to "1." Then, this small filter representation and the encrypted data protected by the Hybrid AES_GCM framework are sent to the cloud server.

To make a secure trap door, the data owner creates a hash of the search phrase with the same hash functions when he or she wants to do a phrase search.

The cloud server uses the Bloom Filter to see if all the bits at the hashed positions are set to "1." In that regard, it is likely that the phrase appears in the dataset. This technique prevents the server to discover the actual keywords or phrases that maintain the privacy of queries and data.

It also lets users search encrypted IoT data quickly and easily.

Hash functions

They have two supported, different configurations (implementation configurable):

HMAC is strongly pseudo-random and is resistant to cryptanalytic attacks; salting with independent keys is process of crypt analysis, which is close to pseudo-randomisations of isolated hash functions.

Non-cryptographic (performance It follows MurmurHash3 using independent seeds (seed = 1..k) in server-side indexing when the operator of the system will tolerate the weaker cryptographic guarantees. Rationale In some cases MurmurHash3 is faster and not cryptographic; it should be used in case the threat model allows it.

Independence approximation. Precise independent hash functions are expensive; we use an approximation of independence by means of an HMAC with dissimilar keys or via doubling (e.g. h i(x) = (h1(x) + ih2(x)) mod m) as suggested by common practice guidelines.

3.5 Phrase Index and Trapdoor Generation

Users can search through encrypted datasets without letting the server or system that holds the data see the search terms (queries) thanks to the trapdoor. The trapdoor allows the system to determine whether there are any records that are identical to the hashed terms in the Bloom filter, however it is not able to determine what the search terms are.

Let a phrase query be represented as in Eq. (6).

$$Q = (t_1, t_2, \dots, t_w) \quad (6)$$

Here, the term t_i denotes the words in the query phrase and *x* represents the phrase length.

To preserve word ordering, the query is converted into ordered n-grams that is given in Eq. (7).

$$G(Q) = \{(t_1, t_2), (t_2, t_3), \dots, (t_{w-1}, t_w)\} \quad (7)$$

A trapdoor is generated using the same hash functions employed during index construction that is given in Eq. (8).

$$T_Q = \{h_j(g) | g \in G(Q), j = 1, \dots, k\} \quad (8)$$

The cloud server compares the trapdoor positions with the corresponding positions in the stored Bloom Filters. A document is considered a potential match if all required bit positions are set to 1.

In the phrase search algorithm, it: Hash and encrypts the query in order to render it safe. It provides an efficient display of query using a Bloom filter. Ciphers the Bloom filter using AES-GCM in order to secure the privacy. This allows searches of encrypted information, with the original query being disclosed. Phrase index construction is given in Algorithm 1 and trapdoor generation is given in Algorithm 2.

Algorithm 1: Phrase Index Construction
Input: Document, Maximum phrase length, Number of hash functions
Output: Bloom Filter
Preprocess document
• Remove stopwords • Normalize text • Tokenize into words
Initialize Bloom Filter of size with all bits set to 0
For $n = 1$ *to* w_{max} do
Generate all contiguous n-grams from Document
For each generated n-gram do
For j = 1 to k do
pos = hj(g) mod m $pos = h_j(g) mod\ m$
BF[pos] = 1
End For
End For
Encrypt Bloom Filter using AES-GCM
Store encrypted Bloom Filter index in the cloud
Return Bloom Filter

Algorithm 2: Phrase Trapdoor Generation and Search
Input: Query phrase Q, Maximum phrase length w_{max}
Output: Ranked matching documents
Tokenize query phrase
If $length\ (Q) \leq w_{max}$
Generate ordered n-gram representation
Else
Divide Q into overlapping windows of size w_{max}
End If
For each phrase window do
For j = 1 to k do
Compute trapdoor position $h_j(g) mod\ m$
End For
End For
Construct query Bloom Filter

Compare query Bloom Filter with stored Bloom Filter indexes
Compute relevance score
Rank documents according to relevance score
Return top matching encrypted documents

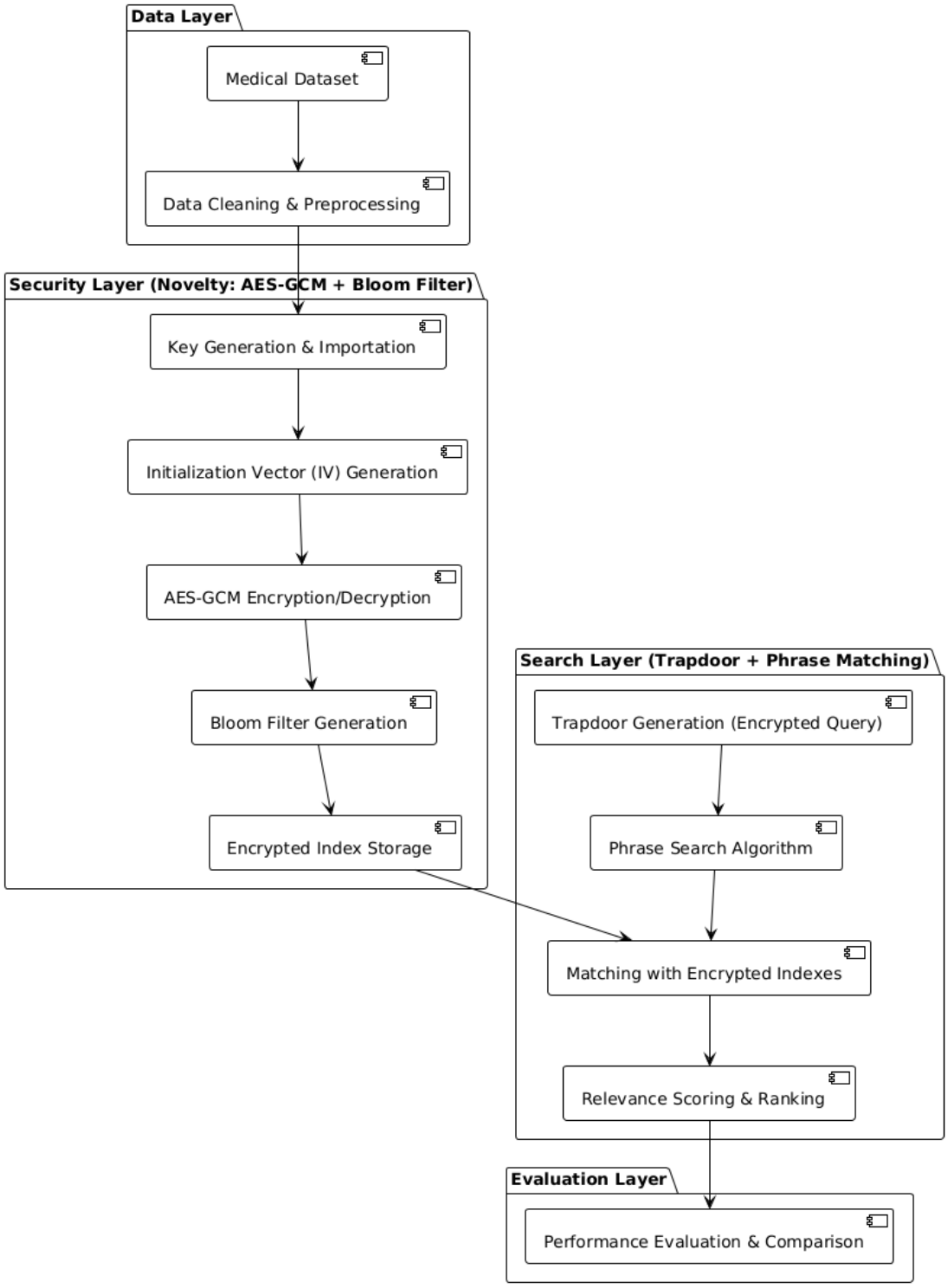

Fig. 1 Architectural diagram for proposed methodology

3.6 Development of Phrase Search Algorithm

- One of the functions converts the indexbloomfilterencrypted with the aeskey provided in the AES-GCM mode.
- Probably the decrypted Bloom filter (indexbloomfilter) is in its form of a byte.

- The Bloom filter (decrypted) and the trapdoor filter (already encrypted) are transformed into bitarrays that are easily compared.
- Bitarray: There is a compact and efficient data structure that is used to represent binary data, which is called bitarray.
- Calculates the number of matching 1s i.e the common elements between the Bloom filters (matchcount).
- Combinations Unions of the two Bloom filters Counts the number of unique 1s (totaluniqueones).

Calculates the relevance score which is the number of bits matching divided by the number of unique bits. To rank the retrieved documents, the relevance score is calculated as in Eq. (9).

$$\text{Relevance Score} = \frac{\text{Match Count}}{\text{Total Unique Ones}} \quad (9)$$

The relevance score ranges from 0 to 1, where a value closer to 1 indicates a stronger similarity between the query phrase and the indexed document.

- A relevance score is obtained between 0 and 1, where:
  - 0: No similarity between the Bloom filters.
  - 1: Perfect match (all bits align).

4. Scheme Construction

The proposed system architecture has three main parts: the Data Owner (DO), the Cloud Server (CS), and the Data User (DU). They are all critical towards ensuring that the data is stored securely, searched fast and accessed by only authorized individuals.

Data owner cleanses the raw data, splits it into useful documents and locates the most significant phrases to use on future searches. The data owner makes Bloom Filters for each document before sending the data to the cloud. These Bloom Filters index those keywords and phrases that have been selected with the help of a probabilistic data structure. This is what makes membership testing quick and it does not occupy space. The data owner uses a Hybrid AES_GCM encryption scheme to protect the real data. The AES_GCM encrypts messages and authenticates them, which ensures that the data is secure and confidential. A reliable key management system, or an already built secure channel, allows the genuine data consumers to exchange the symmetric key safely.

The data owner uploads both the encrypted files and the Bloom Filter indexes to the cloud server after they have been encrypted and the indexes have been made.

Cloud Server resembles a third-party storage and computing product. It retains the cipher-text and the Bloom Filter indexes which accompany it.

When an authorized data user sends a search request to the cloud server, it uses Bloom Filters to do a quick phrase search without decrypting any data. Since Bloom Filters are probabilistic, the server is able to swiftly determine whether a given phrase is present in a document by testing the hash functions which accompany it.

If the filter says there might be a match, the cloud server gets the encrypted files that are relevant and sends them to the data user.

The cloud server runs on an honest-but-curious model, which is very important. It executes search queries on request although it is unable to learn anything significant about the plaintext data or the query made by the user.

The data user makes a trapdoor (an encrypted query) by using the same hash functions that were used to make the Bloom Filter. This trapdoor is sent to the cloud server so that it can be matched. Bloom Filters make it easy and fast to match phrases, and Hybrid AES_GCM makes sure that the system is very secure with very little impact on performance. The main problem is how to make it possible to search for phrases in large amounts of IoT data in untrusted cloud environments while still protecting privacy.

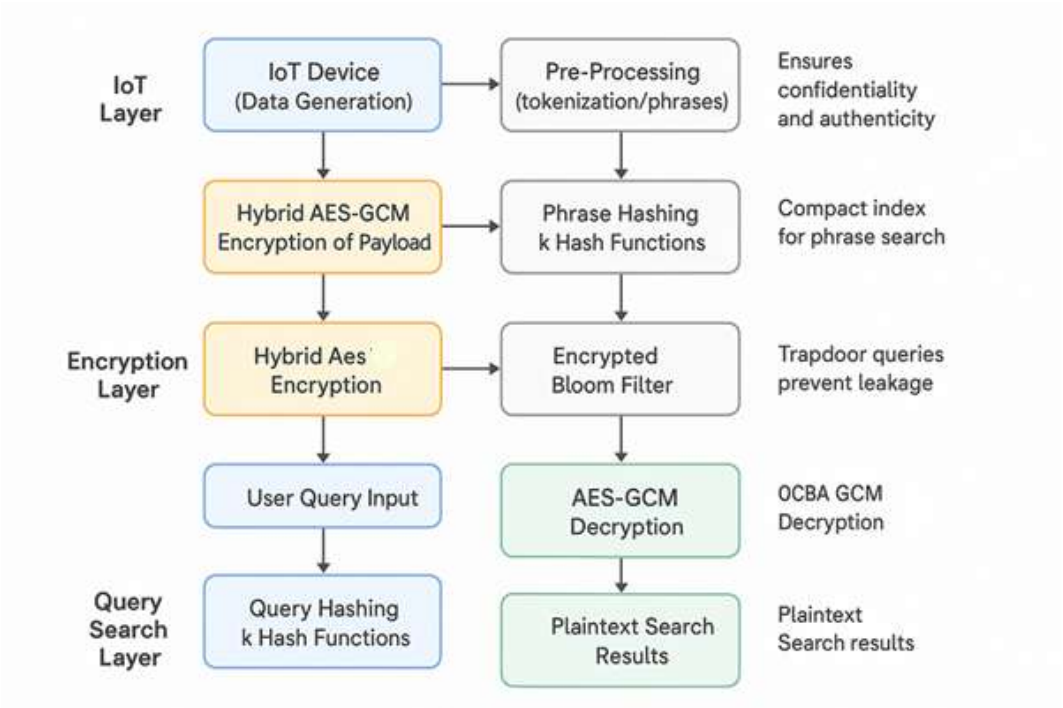

Fig. 2 Flow diagram for scheme construction

5. Security and Privacy Analysis

5.1 Threat Model

The proposed privacy-preserving phrase search framework is based on the honest-but-curious cloud server model. Let A be an adversary who tries to learn sensitive information from the encrypted IoT data, Bloom Filter indexes, and search performed by users. The adversary can be assumed to have the ability to view the medical record, which is assumed to be encrypted, stored in the cloud, monitor the index, which is encrypted in a Bloom Filter, generated when performing a phrase search, and analyse the returned sets and requests for a phrase search. Also, the adversary can execute a statistical analysis and a frequency-based inference attack on the observed information. The adversary, however, does not have the secret AES-GCM key, the keys of the hash functions used in the construction of the Bloom Filter, or access to the medical records or search queries. In this threat model, the adversary aims to steal the sensitive keywords, guess the user's search intent, or extract the plaintext of a medical record embedded with cryptographic keys, without having the appropriate keys.

5.2 Data Privacy and Query Protection

The suggested framework tackles various security and privacy issues with storing sensitive medical data generated via IoT devices in untrusted cloud environments. AES-GCM authenticated encryption is used to ensure data confidentiality, offering encryption and integrity protection. AES-GCM simultaneously encrypts the medical records before sending them out to the cloud and also produces an authentication tag for verifying the integrity of the medical records when they are retrieved. Therefore, if an attacker gets access to this cloud storage, the medical records will still be unreadable if their encryption key is not known. In addition, in the course of the authentication verification process, any record that has been modified without permission will be noticed as modified. Phrase searching is done using a trapdoor technique to maintain query privacy. The user will not send the search word to the cloud server, but instead he/she will create a trapdoor using the same hash functions used for the construction of the Bloom Filter. The cloud server only sees the trapdoor representation and cannot see the plaintext query. The Bloom Filter indexing also hides the keywords, by encoding phrases as bit positions in a bit vector. This means that the contents of the document and user queries are unaffected during the search process.

5.3 Resistance to Security Attacks

With the proposed scheme, since the indexed phrases are not stored in plain text, the keyword guessing attacks are thwarted. All phrases are converted to Bloom Filter representation and secured with cryptographic hashing and AES-GCM encryption. If an adversary does not have the secret keys, he or she cannot efficiently recover the keyword positions from the Bloom Filters, and so cannot efficiently recover the keywords themselves. The framework also prevents using frequency analysis attacks. The cloud server can only see encrypted data, encrypted indexes, and trapdoor representations, and thus cannot see the frequency of certain keywords or phrases in the data. The Bloom Filter indexing, and hashed phrase representation hides the connection between the bit patterns that occur in the bloom filter and what the actual search terms are, which makes statistical inference attacks less effective. In terms of access-pattern

leakage, contents of documents are not disclosed during search operations using the proposed system as phrase matching is done through encrypted Bloom Filter indexes. Like many real-world searchable encryption schemes, however, the cloud server can see which of the encrypted documents are returned as a result of a search query. The contents of the document are still secured, but some data about how the results of the document are accessed can be transferred. The framework also minimizes the leakage of search patterns since the query terms in plain context are never sent to the cloud server. Trapdoor generation hides key terms that are being searched and thus obfuscates user interests. However, if the same query is repeated several times, only partial information about query repetition may be obtained, a frequent problem with searchable encryption systems. The authenticated encryption property of AES-GCM helps to prevent replay attacks. There is an authentication tag attached to each encrypted record that is checked when decrypting the record. If a ciphertext is replayed, modified or tampered, the authentication verification process is not successful and the system rejects it. This mechanism ensures data integrity and authenticity during storage and retrieval operations. In summary, AES-GCM authenticated encryption, Bloom Filter based indexing, and the search-pattern disclosure and replay attacks resistance offered by a trapdoor allows query privacy, query confidentiality, integrity and mitigates the threats of keyword inference, frequency analysis and search-pattern disclosure in cloud-based IoT systems.

6. Experimental Results and Analysis

6.1 Experimental Setup

The proposed framework for the phrase search with privacy was implemented with the help of the Python programming language, where AES-GCM was used as a authenticated encryption algorithm and the BloomFilter was used as a phrase indexing algorithm. Experiments were run on datasets of different sizes of indexes from 10,000 to 50,000 records. The indexes generation time, the trapdoor generation time and the phrase search time are assessed. The processor power for the proposed system was a 2.80 GHz 6-core processor, while the baseline implementation used a 12-exec 3.30 GHz 6-core processor. Table 3 shows the size of the Bloom Filter, number of hash functions, maximum phrase length, and length of the AES-GCM key and authentication tag and other settings of the proposed system.

Table 3. Configuration Parameters of the Proposed Framework

Parameter	Value
Bloom Filter Size (m)	2048 bits
Number of Hash Functions (k)	7
Maximum Phrase Length	5 words
Number of Indexed Phrases (n)	Dataset-dependent
Phrase Generation Method	Contiguous n-grams (1–5 grams)
Encryption Scheme	AES-GCM Authenticated Encryption
AES Key Length	256 bits
Authentication Tag Length	128 bits
Search Mechanism	Trapdoor-based Phrase Search
Index Type	Encrypted Bloom Filter
Query Processing	Phrase-aware Trapdoor Generation
Relevance Measure	Bloom Filter Similarity Score

6.2 Index Generation Performance

Table 4 and Figure 3 show the index generation time on different number of indexes. For both methods, the index generation time is almost linear with the size of the dataset. The proposed technique needs 0.093 s for 10,000 indexes and 0.344 s for 50,000 indexes, whereas the existing technique needs ranging from 0.010 s to 0.100 s. The proposed method has a slight overhead, but this is mainly for AES-GCM authenticated encryption and building the PhraseOriented Bloom Filter. The results show that the indexing process is still computation friendly even for the large number of data points. The near-linear increase in growth indicates good scalability properties of the proposed scheme, and that the proposed scheme is appropriate for large-scale IoT healthcare repositories, without causing excessive indexing delays.

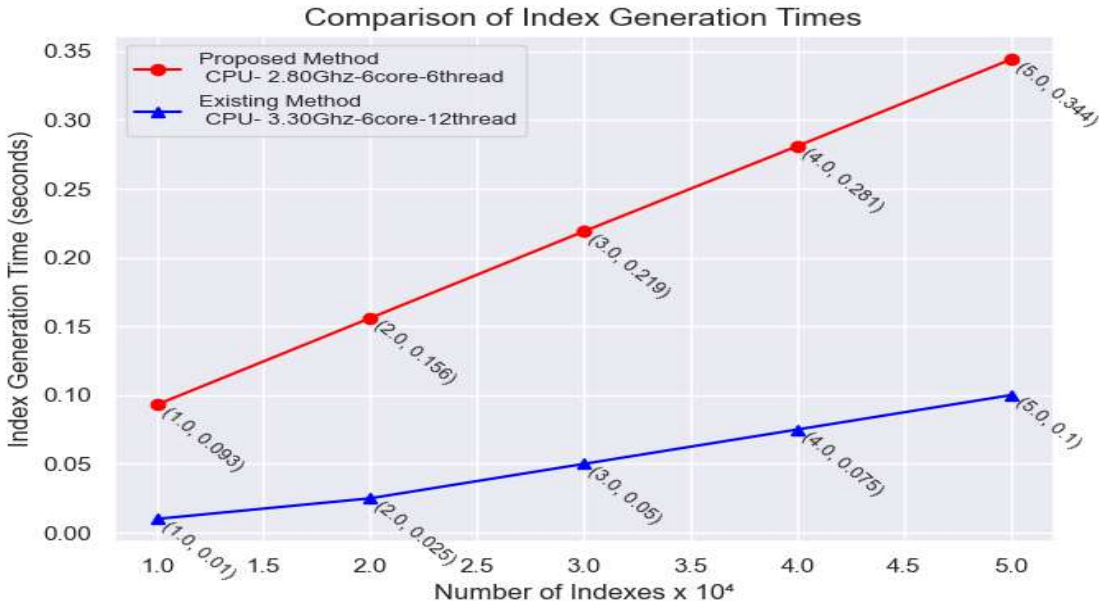

Fig. 3 Index generation time for varying numbers of encrypted Bloom Filter indexes using AES-256-GCM authenticated encryption and a maximum phrase length of 5 words

Table 4. Index Generation Time Comparison

Number of Indexes (×10⁴)	Proposed Method (s)	Existing Method (s)
1	0.093	0.01
2	0.156	0.025
3	0.219	0.05
4	0.281	0.075
5	0.344	0.1

6.3 Trapdoor Generation Performance

The results of the trapdoor generation time comparison are presented in Table 5 and Figure 4. Approximatively 0.001ms required by proposed method is less than 0.002ms required by existing method for generating the trapdoors. This means that the generation time of the traps is reduced by about 50%. The lightweight phrase hashing mechanism and efficient Bloom Filter representation used in the proposed framework is responsible for the improvement. In real-time healthcare IoT applications, such as systems, users need to access data quickly and with minimal latency, making fast generation of traps crucial.

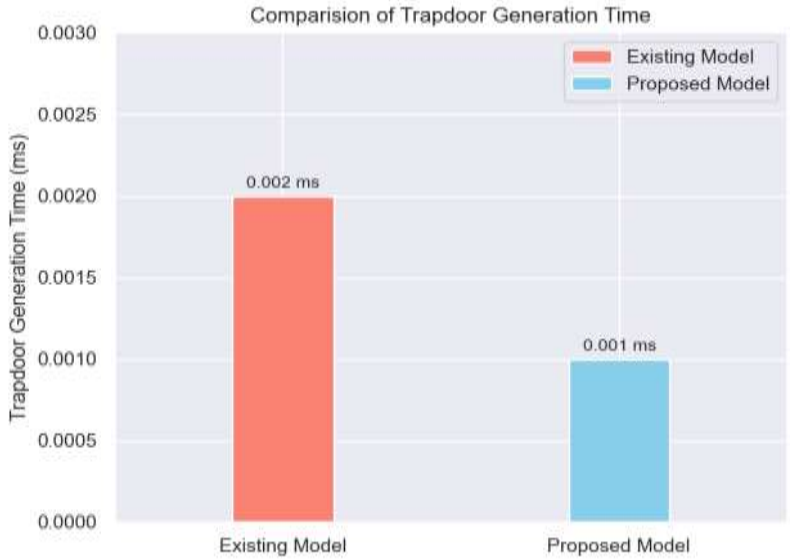

Fig. 4 Trapdoor generation time measured under the proposed encrypted Bloom Filter phrase-search framework using AES-256-GCM authenticated encryption.

Table 5. Trapdoor Generation Time Comparison

Method	Trapdoor Generation Time (ms)
Existing Method	0.002
Proposed Method	0.001

6.4 Phrase Search Performance

The search times for the phrases are compared with different index sizes in Table 6 and Figure 5. The search time of the proposed scheme increases from 78 s to 156 s as the number of indexes goes from 10000 to 50000. The time is from 20 s to 100 s with the current method. The proposed approach has a higher search latency but at the same time offers phrase-level search, query privacy, authenticated encryption and secure Bloom Filter matching. The results show that the extra security mechanisms added to the computational overhead is still acceptable for practical cloud-IoT deployments.

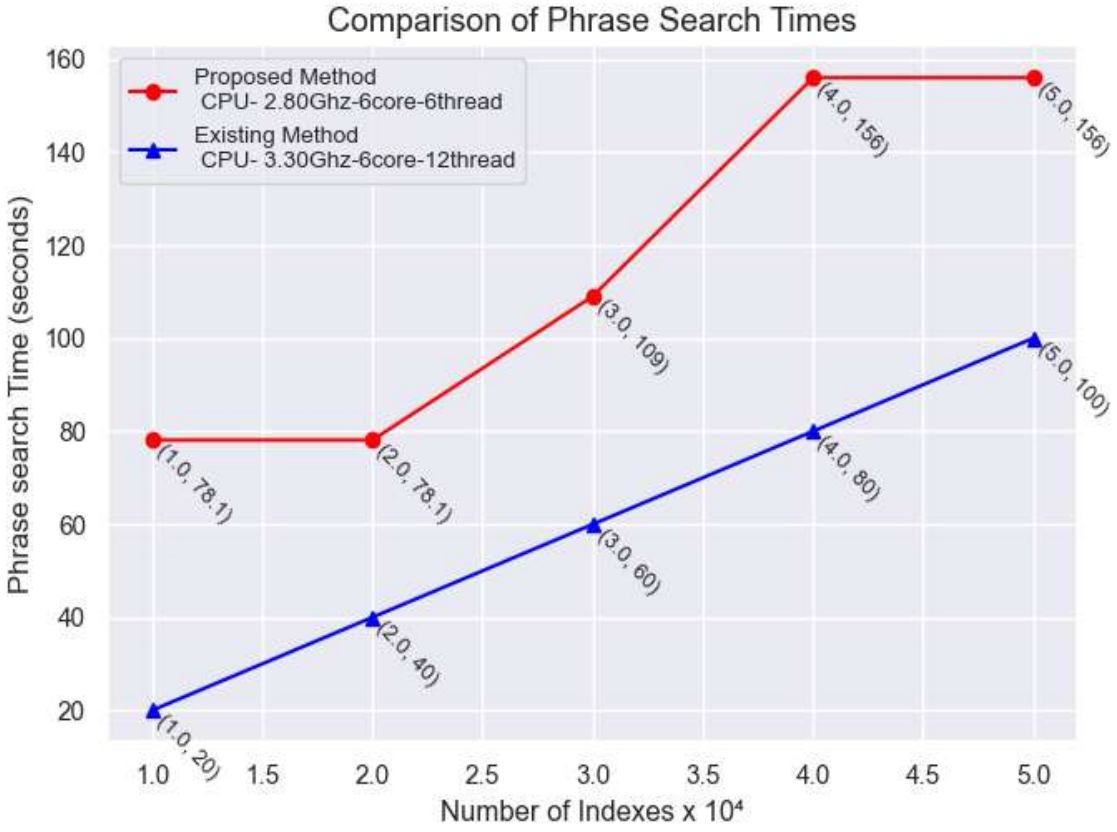

Fig. 5 Phrase search time for different index sizes under the proposed privacy-preserving search framework based on encrypted Bloom Filters and AES-256-GCM authenticated encryption.

Table 6. Phrase Search Time Comparison

Number of Indexes (×10⁴)	Proposed Method (s)	Existing Method (s)
1	78	20
2	78	40
3	109	60
4	156	80
5	156	100

6.5 Relevance Score Analysis

The proposed framework does not carry out any document classification thus not calculating classification accuracy. In this context, effectiveness of retrieval is evaluated by a relevance score based on the similarity between query Bloom Filter and the stored encrypted Bloom Filter indexes. The proposed framework does not assess the accuracy of the predictions, as it does not have a classification-based system. On the contrary, the effectiveness in retrieving is evaluated by a relevance score based on the Bloom Filter similarity. The relevance score is used for measuring the score of query trapdoor and stored encrypted indexes. Scores

vary from 0 to 1, being closer to 1 is the more similar that the query string is to the retrieved document. This metric allows ranking of the retrieved records based on estimated relevance without compromising search privacy.

6.6 Discussion and Limitations

The experimental results show the effectiveness of the proposed framework in preserving privacy while maintaining search capabilities in a cloud-based IoT environment. The framework not only offers data confidentiality, query privacy, data integrity protection, and phrase-search, but also brings extra computation burden for the integration of AES-GCM authenticated encryption and phrase-aware Bloom Filter indexing. The observed trends in the performance results validate the scalability of the proposed solution and the potential for its application in privacy-sensitive IoT-based healthcare systems. The observed trends in the performance results are indicative of the scalability of the proposed approach and its applicability to privacy-sensitive healthcare IoT applications.

The results reported in this paper must be interpreted with regard to the hardware and software setup in which the experiments are executed. In the future work, we would like to experiment the proposed system and benchmarking systems in the same hardware, same dataset and same operating system, so that the comparing result can be fair enough. The performance results are presented in this paper only as suggestive results toward understanding the computation performance characteristics of the suggested approach, but not as rigorous benchmark results. Although the overhead due to the use of authenticated encryption (AES-GCM) and encrypted Bloom filter for phrase indexing is incurred in the suggested framework, the time execution of the system shows acceptable results while offering greater security and protecting data confidentiality, data integrity, and query privacy.

7. Conclusion and Future Scope

The proposed approach ensures strong protection of sensitive IoT data stored in the cloud while enabling fast and efficient phrase-based search. The system uses Bloom Filters to index quickly at low storage overhead and fewer searches, and limit false positives on a search query. AES-GCM Authenticated Encryption is also integrated, which increases the level of security and provides additional protection against sophisticated cyber threats by providing authenticated encryption, protecting confidentiality and integrity of data. This architecture provides a good balance between security, search performance and scalability, and it is therefore very applicable to applications of resource-constrained Internet of Things that rely on third-party cloud computing infrastructure.

Future work: The proposed privacy-preserving phrase search framework offers several improvements that are interesting for future research. Multi-keyword search and ranked search can be added to handle more complicated queries and enhance the retrieval quality. Additionally, advanced privacy-preserving methods like homomorphic encryption and secure multi-party computation could be leveraged for secure processing of the encrypted data even if the data is not decrypted, and light-weight cryptographic primitives could be considered to reduce the computational load on resource-limited IoT devices. It can be tailored for edge and fog computing use cases to eliminate latency and bandwidth limitations and minimize processing cost in distributed IoT deployments and even include blockchain-based access control to bolster auditability and trust. As the current framework does not classify documents, but rather retrieve phrases and rank them by relevance, the application of the confusion matrix is not suitable and future work will use benchmark datasets that contain ground-truth relevance annotations to assess the effectiveness of retrieval on the basis of standard Information Retrieval metrics like Precision, Recall and F1-score. Moreover, controlled benchmarking experiments will be performed on the same hardware platforms, datasets and operating environments to provide a fair comparison with the existing methods. There will also be large-scale scalability testing of datasets with sizes ranging from 10K to 500K to assess the search latency, memory usage, index size, and storage efficiency. Such improvements will further bolster the flexibility, scalability, security and real-world usability of privacy-preserving searchable encryption in a cloud-IoT setting.

References

1. Ali, M., Basha, I., & Qureshi, K. N. (2023). A lightweight privacy-preserving phrase search using compressed Bloom Filters for IoT. Computer Networks, 226, 109697. https://doi.org/10.1016/j.comnet.2023.109697
2. Chen, H., Li, J., & Zhang, Y. (2021). Dynamic and privacy-preserving phrase search over encrypted IoT data in cloud environments. IEEE Internet of Things Journal, 8(12), 9802–9814. https://doi.org/10.1109/JIOT.2021.3062391
3. Chen, Y., Wang, T., & Sun, H. (2022). Secure and efficient multi-phrase search over encrypted IoT data in cloud-assisted edge computing. *IEEE Internet of Things Journal*, 9(5), 3945–3957.

https://doi.org/10.1109/JIOT.2021.3106542
4. Elhoseny, M., & Riad, K. (2025). RT-PPS: Real-time privacy-preserving scheme for cloud-hosted IoT data. *Journal of High-Speed Networks, 31*(1). https://doi.org/10.3233/JHS-240096
5. Gao, Y., Yu, J., & Wang, T. (2023). Fine-grained privacy-preserving phrase search with verifiable result correctness for IoT data. *Information Sciences*, 628, 199–214. https://doi.org/10.1016/j.ins.2023.02.057
6. He, D., Kumar, N., & Khan, M. K. (2021). Privacy-aware searchable encryption for IoT-based big data storage. *Future Generation Computer Systems*, 124, 257–269. https://doi.org/10.1016/j.future.2021.05.009
7. Kumar, R., & Das, M. L. (2021). An efficient AES-GCM based lightweight data protection scheme for IoT devices. Journal of Information Security and Applications, 59, 102842. https://doi.org/10.1016/j.jisa.2021.102842
8. Liu, S., & Chen, X. (2020). Privacy-preserving phrase search using secure positional indexing. Future Generation Computer Systems, 108, 1017–1026. https://doi.org/10.1016/j.future.2020.03.009
9. Park, J., Kim, Y., & Lee, H. (2022). Edge-assisted secure data storage using AES-GCM for IoT applications. Sensors, 22(4), 1372. https://doi.org/10.3390/s22041372
10. Peng, T., Gong, B., Tu, S., Namoun, A., Alshmrany, S., Waqas, M., Alasmary, H., & Chen, S. (2024). Forward and backward private searchable encryption for cloud-assisted industrial IoT. *Sensors, 24*(23), 7597. https://doi.org/10.3390/s24237597
11. phrase search over dynamic IoT data streams. *IEEE Transactions on Mobile Computing*, 21(4), 1526–1540. https://doi.org/10.1109/TMC.2021.3051693
12. Rahman, M. A., Hossain, M. S., & Alhamid, M. F. (2022). Privacy-preserving phrase search with edge–cloud collaboration for IoT data. IEEE Transactions on Industrial Informatics, 18(8), 5394–5403. https://doi.org/10.1109/TII.2021.3122367
13. Shiraly, D., Eslami, Z., & Pakniat, N. (2024). *Hierarchical Identity-Based Authenticated Encryption with Keyword Search over Encrypted Cloud Data*. Journal of Cloud Computing, 13, 112. https://doi.org/10.1186/s13677-024-00633-9
14. Singh, A., Sharma, V., & Kumar, N. (2022). A hierarchical Bloom Filter approach for efficient phrase search over encrypted IoT data. IEEE Access, 10, 45678–45689. https://doi.org/10.1109/ACCESS.2022.3165678
15. Wang, T., Li, J., & Li, H. (2021). Privacy-preserving multi-keyword and phrase search using Bloom Filters over encrypted cloud data. Future Generation Computer Systems, 117, 321–333. https://doi.org/10.1016/j.future.2020.12.034
16. Xu, K., Li, M., & Zhou, Y. (2023). Inference-resilient searchable symmetric encryption for secure cloud storage. IEEE Transactions on Dependable and Secure Computing. Advance online publication. https://doi.org/10.1109/TDSC.2023.3245609
17. Yan, X., Zheng, C., Tang, Y., Huo, Y., & Jin, M. (2023). Dynamic forward secure searchable encryption scheme with phrase search for smart healthcare. *Journal of Systems Architecture*, *144*, 103003. https://doi.org/10.1016/j.sysarc.2023.103003
18. Yang, Y., Lin, X., & Ma, J. (2020). An efficient and privacy-preserving multi-keyword ranked search scheme over encrypted cloud data. *IEEE Transactions on Cloud Computing*, 10(2), 1103–1116. https://doi.org/10.1109/TCC.2020.2996763
19. Zhang, F., Xia, X., Gao, H., Ma, Z., & Chen, X. (2025). *A Blockchain-Enabled Multi-Authority Secure IoT Data-Sharing Scheme with Attribute-Based Searchable Encryption for Intelligent Systems*. Sensors, 25(19), 5944. https://doi.org/10.3390/s25195944
20. Zhang, P., Chen, X., & Xu, K. (2021). Secure phrase search over encrypted cloud data using structured indexes. IEEE Transactions on Services Computing, 14(6), 1754–1767. https://doi.org/10.1109/TSC.2020.2984965
21. Zhou, Y., Tang, B., & Yang, Y. (2024). *A Lattice-Based Searchable Encryption Scheme with Multi-User Authorization for the Certificateless Cloud Computing Environment*. Transactions on Emerging Telecommunications Technologies, 35(4), e4960. https://doi.org/10.1002/ett.4960