IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 September 2023, accepted 10 October 2023, date of publication 16 October 2023, date of current version 23 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3324817

RESEARCH ARTICLE

Privacy Preserving and Serverless Homomorphic-Based Searchable Encryption as a Service (SEaaS)

MUSFIRAH IHTESHAM[1], SHAHZAIB TAHIR[2], (Senior Member, IEEE), HASAN TAHIR[1], (Senior Member, IEEE), ANUM HASAN[2], AIMAN SULTAN[2], SAQIB SAEED[3], AND OMER RANA[4], (Senior Member, IEEE)

[1]School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
[2]College of Signals, National University of Sciences and Technology (NUST), Rawalpindi 44000, Pakistan
[3]Saudi Aramco Cybersecurity Chair, Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam 31441, Saudi Arabia
[4]Physical Science and Engineering College, Cardiff University, CF10 3AT Cardiff, U.K.

Corresponding author: Shahzaib Tahir (shahzaib.tahir@mcs.edu.pk)

This work was supported by the Engineering and Physical Sciences Research Council funded Project PACE: Privacy-Aware Cloud Ecosystems under Grant EP/R033293/1 and Grant EP/R033439/1.

ABSTRACT Serverless computing has seen rapid growth, thanks to its adaptability, elasticity, and deployment agility, embraced by both cloud providers and users. However, this surge in serverless adoption has prompted a reevaluation of security concerns and thus, searchable encryption has emerged as a crucial technology. This paper explores the Searchable Encryption as a Service (SEaaS) and introduces an innovative privacy-preserving Multiple Keyword Searchable Encryption (MKSE) scheme within a serverless cloud environment, addressing previously unmet security goals. The proposed scheme employs probabilistic encryption and leverages fully homomorphic encryption to enable operations on ciphertext, facilitating searches on encrypted data. Its core innovation lies in the use of probabilistic encryption for private multi-keyword searches. To validate its practicality, we deploy the scheme on the public cloud infrastructure, ''Contabo,'' and conduct rigorous testing on a real-world dataset. The results demonstrate that our novel scheme successfully preserves the privacy of search queries and access patterns, achieving robust security. This research contributes to the field of serverless cloud security, particularly in the context of searchable encryption, by providing a refined solution for safeguarding data while maintaining usability in a serverless computing landscape.

INDEX TERMS Homomorphic encryption, cloud computing, multi-keyword search, serverless computing, trapdoor generation, search pattern security.

I. INTRODUCTION

Cloud computing has enabled convenient usage of computational and network resources [1]. For data storage as well as its management, the cloud provisions multiple services. The cloud is inexpensive [2] but storing data on the Cloud makes it vulnerable to advanced persistent threats (APT) [3]. Serverless computing is rapidly expanding as a result of its widespread adoption by cloud providers and tenants due to its adaptability, elasticity, and deployment agility With the boom in the serverless paradigm, security issues are being reevaluated as well [4], [5]. One way to protect data from these vulnerabilities is by encrypting it prior to its storage on the cloud. However, traditional encryption techniques limit the usability of the data *i.e.* keyword search can not be performed on the ciphertext. The solution to this lies in searchable encryption (SE).

Homomorphic Encryption (HE) is a groundbreaking approach that has enabled cloud data owners/users to not only store but also to search data [6]. HE paired with

The associate editor coordinating the review of this manuscript and approving it for publication was Nitin Gupta.

SE has opened many pathways for the cloud computing industry. Consider the healthcare industry, which has shifted towards adopting serverless cloud solutions to cater to the high volumes of patient records. Patient data consists of sensitive health-related records whereby the confidentiality and privacy of the patient data are of utmost importance [7], [8], [9]. SE enables secure searching on the encrypted patient data [10], [11]. Similarly, in the aviation industry, SE plays an important role in protecting the passenger's personal details while the airline receives verifiable authentication of the passenger's credentials. The latest research [12] has shown the need for SE to solve a variety of real-world problems ranging from the health records of patients to highly critical records in the defense sector.

The first SE scheme with a keyword search on encrypted data was proposed in [13]. Further research performed on SE was described in [14], [15], [16], [17], and [18]. However, the focus of these schemes was single keyword search with Boolean search results. Multiple schemes based kNN, and fuzzy search are presented in [19], [20], [21], and [22]. Ranked multi-keyword search was discussed in [23], [24], [25], [26], and [27], however, it included the use of search index which not only affected the efficiency of search but also diminished the security of the scheme. Formal definitions for SE schemes over cloud computing such as search pattern, access pattern, and adaptive and non-adaptive indistinguishability have been proposed in [15] and [28].

Recent research on SE can be characterized as either index-based search or linear search. Major research has been conducted on index-based SE schemes [29], [30], [31], [32]. Due to the fact that the index structure in index-based search engine schemes reveals details about the correspondence between a given keyword and its associated document, these techniques are intrinsically vulnerable in terms of security. Therefore, the probability of data security breaches is higher in such schemes.

In addition to the security challenges presented by the introduction of index tables, research on SE does not provide search patterns and access pattern privacy. Search pattern privacy is concerned with the identification of keywords given an encrypted trapdoor. Search pattern attack entails the adversary being able to identify which keywords are being searched by the client. while access pattern privacy, on the other hand, is related to the mapping of an encrypted trapdoor consisting of keywords, to a set of documents that contain that respective trapdoor. If the adversary is unable to map a trapdoor to the set of documents containing the trapdoor, access pattern privacy is preserved.

In today's digital landscape, safeguarding sensitive information from ever-evolving security threats is paramount. The need for a cloud security solution to achieve high levels of security and performance forms the motivation of this research. This research seeks to address this challenge by developing a robust searchable encryption solution. By incorporating probabilistic trapdoors, elevating security levels, and optimizing performance efficiency, this work aims to provide search pattern and access pattern security, ensuring the confidentiality and integrity of critical data in an increasingly vulnerable digital world.

This research proposes a novel multi-keyword SE (MKSE) scheme that preserves privacy while enhancing usability in terms of data stored over a serverless cloud. The proposed novel MKSE scheme is based on HE to enable search operations on encrypted text. It provides a high level of security using probabilistic encryption to generate encrypted data and queries. The proposed scheme has been deployed and tested on the ''Contabo'' public cloud. MKSE scheme can have a profound impact across different sectors which include healthcare, aviation, IoT, banking and finance, law enforcement, and education.

A. CONTRIBUTIONS

This research contributes the following:

- A novel privacy-preserving multi-keyword SE (MKSE) scheme based on fully HE is presented which provisions data confidentiality and privacy over a serverless cloud environment. The scheme is strategically based on probabilistic trapdoors thus hiding search patterns.
- The proposed homomorphic capability reduces the client-side computations and the leakages associated with the conventional SE schemes. A thorough security analysis verifies the security definitions. The proof of concept prototype is deployed and tested over the cloud platform ''Contabo'', to analyze its performance.

B. ORGANIZATION

The rest of this paper is organized as follows: Section II reviews existing research on multi-keyword SE. Section III presents the cryptographic primitives (DGHV) used in the MKSE scheme. The system overview elaborates on the system model, the threat model, as well as the security goals in Section IV. Section V revisits the security definitions associated with the proposed scheme. The proposed MKSE scheme is presented in detail in section VI. The security analysis highlighting leakage profiling is carried out in Section VII. Implementation aspects are described in Section VIII while the performance evaluation presenting the computational analysis and discussing the limitations is carried out in Section IX. Section X summarises the research and concludes with possible future directions.

II. RELATED WORK

SE is an important cryptographic primitive that provisions search on encrypted data stored on the cloud. It is a revolutionary algorithm that lets entities perform search queries on the ciphertext. This section discusses existing literature on multi-keyword SE as well as some of the latest works in serverless cloud computing.

Secure and dynamic multi-keyword SE is proposed in [31]. The authors have used tf-idf and vector space model to generate an index table and queries. The secure k-NN approach is used to encrypt documents and queries.

They have employed relevance scores between the index and query to provide accurate ranked results to the cloud user. Using transformation matrices, they have provided resistance against statistical attacks.

In [30], the authors have utilized the sparsity of matrix to propose a k-NN based MKSE scheme. They have also reviewed previously proposed SE schemes based on k-NN and bloom filter. Based on the k-NN approach, they have chosen sparse matrix pairs to efficiently encrypt the indices. Further, they have used bloom filters to solve the dictionary update problem. Their proposed scheme provides data privacy, keyword privacy, trapdoor privacy, and access pattern privacy.

Ahamed et al. proposed a lightweight searching protocol for RFID tags in a serverless cloud environment [33]. This protocol can effectively carry out the search for a specific tag without the need to use and maintain server(s). A privacy-preserving MKSE is proposed in [34] using Ciphertext Policy Attribute-Based Encryption (CP-ABE). The authors have addressed the privacy concerns present in the healthcare industry regarding private patient records in multi-owner settings *i.e.* multiple patients. Their scheme also provides trapdoor/keyword privacy. Using real-world datasets they have shown that their scheme is practical and feasible for implementation.

The authors have also used secure k-NN to provide search security in [24]. The scheme provides authorization and ranking for cloud users. Trapdoor unlinkability, confidentiality, and collusion resistance are also enabled for the search on the encrypted data. Bilinear mapping is used in [25] to provide privacy-preserving MKSE for multiple data owners. The scheme has employed the tf-idf model to provide ranked results to cloud users. The tree-based index is used with the construction of a privacy-preserving function to provide efficient search results for every data owner. These indexes are merged by the cloud server. Then, depth-first search is used in this scheme to find required top-k files to return to the data user.

In mobile cloud computing settings, a new method SEED was suggested [35] for serverless efficient encrypted deduplication. The authors claimed that SEED guaranteed the security features of data consistency, confidentiality, as well as collusion resistance for the data in cloud without the use of extra servers. The lack of specialized servers improved SEED's efficacy for the mobile cloud, where user movement is deemed critical.

Authors in [26] have proposed an efficient SE scheme for multi-keywords in mobile cloud storage. Their proposed scheme incorporates relevance score and k-NN technique to provide accurate and ranked search results. Furthermore, they have used blind storage to ensure that the access pattern of cloud users is concealed. An efficient index is generated to enable the practical functioning of the scheme. Their scheme also provides data and index confidentiality, trapdoor privacy, and search pattern concealment.

Minhashing-based scheme is presented in [29]. An encrypted index is constructed in multiple steps including feature extraction, index generation using minhash function, and index encryption using HMAC. Encrypted query/trapdoor is generated based on the signatures present in the encrypted index, providing keyword privacy. Ranked results are returned to the cloud users based on relevance score and tf-idf. The authors have further modified this scheme to provide access pattern privacy using two separate servers for searching and retrieval. In the modified scheme, they have used Paillier encryption to encrypt documents and their corresponding relevance scores.

Two Round SE (TRSE) is proposed in [36] using HE and vector space model. Multi-keyword ranked retrieval is performed using relevance score and tf-idf. The scheme provides privacy-preserving SE by performing ranking at the user end. The search pattern is hidden in [37] to provide trapdoor privacy and keyword privacy. The authors have achieved their conjunctive keyword SE using a special variant of additive HE. They have considered two servers *i.e.* a cloud server and an auxiliary server to achieve the privacy goals of their scheme. To augment security measures from the user's perspective, they have employed random polynomials to guarantee that only the desired outcomes can be obtained by the user. Their scheme provides a stronger security guarantee to the cloud user. Their scheme performs an efficient search in parallel, independent of the search index.

HE is used in [38] to enable efficient multi-keyword retrieval through the use of correlation scores to return accurate and ranked results to the cloud user. They have modified a HE scheme to provide secure retrieval of documents. They have shown that their scheme provides keyword privacy and efficient and accurate retrieval of top-k documents.

An ABE-based secure and efficient access control system [39] is developed for serverless security computing for resource and knowledge sharing. The data is first secured using user characteristics before being divided into cyphertext. Finally, it is decoded using a decryption method, and the cyphertext shares are spread across the network, while the encapsulated texts are kept in the serverless system. The authors indicated that the suggested method outperforms the current methods in terms of data security in a serverless system.

Verifiable PKE with keyword search is performed in [40]. The authors have considered that multiple users are accessing the encrypted cloud data through an inverted index. Dual Embedding Space Model (DESM) is presented in [41]. The authors present a lightweight construction aimed at achieving accurate ranked search results. DESM index generation ensures that retrieval of the ranked results is efficient. The authors have proposed a ranked search mechanism enabling multi-keyword search using improved k-NN. They have solved the issue of index updates using dimension reduction in DESM.

A verifiable privacy-preserving MKSE scheme is proposed in [42] to provide verifiable search results to the cloud user. The scheme is based on the adaptive Homomorphic MAC technique to enable verified search results. In this way, the ranked search results returned to the cloud user can be checked and detected for incorrect search results. Using real-world datasets, the authors have performed security and performance analysis of their scheme.

The scheme in [43] put forward a threshold access control for cloud sharing based on groups of data users. A multi-user, MKSE is proposed in [44]. The authors have identified the flaws of the commonly used scheme k-NN and developed a new scheme to eradicate those flaws. Their scheme supports keywords in arbitrary languages and provides flexible authorization and time-controlled revocation of access for cloud users. Another feature provided by their scheme is data privacy protection.

The authors [46] generated a multi-keyword vector to provide searching on encrypted data through probabilistic trapdoors. Their proposed scheme provides data privacy and trapdoor unlinkability and is resistant to indistinguishable attacks, providing enhanced security and functionality to the cloud user. The performance analysis carried out by the authors has shown that their scheme has unique search functionality advantages over other existing multi-keyword schemes.

The paper [47] presents a Single Keyword Searchable Encryption (SKSE) scheme implemented using the Paillier Cryptosystem, with two variants, Secure SKSE and Efficient SKSE. The Secure SKSE scheme prioritizes security through probabilistic encryption and trapdoors, while the Efficient SKSE variant offers significant performance gains, being 84 times faster. The research evaluates both schemes on real-world aviation data in a use-case scenario of airport security, deployed on the Contabo public cloud platform, providing insights into their effectiveness in achieving security and performance objectives.

The authors introduced an innovative approach to multi-keyword ranked searchable encryption (MRSE) in [45], addressing a crucial privacy concern. Unlike previous MRSE methods that solely perform complete keyword searches and ranking on the server-side, MRSW allows users to include a wildcard keyword in their queries, enhancing search flexibility while maintaining data security through a Bloom filter-based approach. The proposed MRSW system is rigorously analyzed for security under adaptive chosen-keyword attack (CKA2) models and demonstrates efficiency and practicality through experiments on real web of science data.

The article [48] addresses critical security concerns in the context of Public Key Encryption with Keyword Search (PEKS) for cloud data storage. It highlights vulnerabilities, such as keyword guessing attacks, incorrect results from untrusted cloud servers, and the looming threat of quantum attacks. To counter these challenges, the paper introduces VR-PEKS, a novel ciphertext retrieval scheme based on fully homomorphic encryption (FHE). VR-PEKS not only enables verifiable searches but also mitigates risks through the use of an oblivious pseudorandom function to randomize keywords and FHE for encryption. Moreover, the article demonstrates the scheme's security and effectiveness, proving its resilience against adaptive keyword selection attacks.

Another scheme introduces the Verifiable SE Framework (VSEF) [49] as a foundational solution that can withstand insider KGA and enable verifiable searches. Building upon this framework, the enhanced VSEF is presented, designed to support multi-keyword search, multi-key encryption, and dynamic data updates. The research underscores the importance of practicality and scalability in real-world SE applications. Extensive experiments with the Enron email dataset demonstrate that the enhanced VSEF achieves both high efficiency and robust resistance to insider KGA while ensuring the verifiability of search results.

Presented studies confirm that there is a need to address privacy-preserving goals such as search, and access pattern privacy, along with trapdoor unlinkability. The presented scheme is proposed to address these privacy concerns using probabilistic encryption. Table 1 emphasizes that the MKSE scheme uses HE to provide security goals such as search pattern privacy, access pattern privacy, and trapdoor unlinkability. In comparison to other schemes, the MKSE scheme provides the most accurate results but does not support ranking since it is not index-based. Table 1 presents the literature review in a summarized form. The security goals presented in the table are briefly described below:

- **Search pattern privacy** is concerned with identifying keywords given an encrypted trapdoor. If the adversary can identify which keywords are being searched by the client, leakage of search pattern privacy exists.
- **Access pattern privacy** is termed as the relation of mapping of an encrypted trapdoor consisting of keywords, to a set of documents that contain that respective trapdoor. If the adversary is unable to map a trapdoor to the set of documents; access pattern privacy is preserved.
- **Trapdoor unlinkability** looks at the link between trapdoor and corresponding keywords. The adversary should not be capable of deducing any link between a keyword and the trapdoor generated by it.

III. PRELIMINARIES

A. DGHV

DGHV [50], is a FHE scheme over integers. The parameters include γ - the bit length of integers in the public key, η - the private key length, ρ - noise length and τ - public key integers. The basic phases of DGHV [50] are described as follows:

1) **KeyGen** (1^{λ})
   The public/private key pair is generated in this phase. For the private key S_k, a random prime integer p of size η bits is generated. For the public key P_k, a random odd integer q_0 is picked such that $q_0 \in [0, 2^{\gamma}/p)$. Then x_0 is calculated using $x_0 = q_0 \cdot p$. Further, a PRNG

TABLE 1. Comparative analysis of multi-keyword searching schemes.

Technique Used	Index based	Ranked	HE based	Search Pattern	Access Pattern	Trapdoor Unlinkability
Ciphertext-Policy Attribute-Based Encryption and SSE [24]	✓	✓	X	X	X	✓
Tree-based Ranked MKSE for Multi-Data Owners [25]	✓	✓	X	X	X	✓
Relevance score and k-NN based SE scheme [26]	✓	✓	X	✓	✓	✓
Lightweight SKSE [28]	✓	X	X	✓	X	✓
Minhashing-based SE [29]	✓	✓	✓	X	X	✓
kNN-based MKSE scheme using Sparse Matrices [30]	✓	✓	X	X	X	✓
Secure k-NN and Greedy Depth-first based Search Scheme [31]	✓	✓	X	X	X	✓
Ciphertext- Policy Attribute-Based Encryption Scheme [34]	✓	X	X	X	X	X
Vector space model and HE based top-k keyword retrieval scheme [36]	✓	✓	✓	✓	✓	X
Special Additive HE Scheme [37]	✓	X	✓	✓	✓	✓
HE-based MK Retrieval Scheme [38]	✓	✓	✓	X	X	X
DGHV and Inverted Encryption Index Structure [40]	✓	X	✓	X	X	✓
Dual Embedding Space Model and kNN [41]	✓	✓	X	X	X	✓
Adapted Homomorphic MAC for MKSE [42]	✓	✓	✓	X	X	✓
Threshold MKSE [43]	✓	X	✓	X	X	X
PHE Threshold Decryption Based MKSE on Arbitrary Language [44]	✓	✓	✓	X	✓	X
Ranked MKSE with Wildcard KW [45]	✓	✓	X	✓	✓	X
MKSE Scheme based on Probability Trapdoor [46]	✓	X	X	X	X	✓
ABE based secure and efficient access control system [39]	X	X	X	X	X	X
Paillier HE-based SKSE scheme [47]	X	X	✓	✓	X	✓
VR-PEKS [48]	✓	X	✓	X	X	X
Verifiable SE Framework [49]	✓	X	X	X	X	X
Proposed Scheme FHE-based MKSE scheme	X	X	✓	✓	✓	✓

is initialized with a random seed value. Using $f(se)$, a set of integers is generated such that $\mathcal{X}_i \in [0, 2^{\gamma})$ for $1 \leq i \leq \tau$. Compute $\delta_i =< \mathcal{X}_i > +\xi_i \cdot p - r_i$ where $r_i \longleftarrow \mathbb{Z} \cap (-2^{\rho}, 2^{\rho})$ and $\xi \longleftarrow \mathbb{Z} \cap [0, 2^{\lambda+\eta}/p)$. Public key is set as $P_k = (se, x_0, \delta_1, \ldots.., \delta_\tau)$.

2) **Encrypt** $(P_k, m \in 0, 1)$
Using the pseudorandom generator $f(se)$, the integers $\mathcal{X}_i$ are recovered and x_i is calculated such that $x_i = \mathcal{X}_i - \delta_i$ for $1 \leq i \leq \tau$. A random integer $r \in (-2^{\rho'}, 2^{\rho'})$ and vector b is chosen such that $(b_i) \in [0, 2^{\alpha}]^{\tau}$. Then, the ciphertext is calculated using:

$$c = 2r + m + 2\sum_{i=1}^{\tau}(b_i \cdot x_i)_{x_0}$$

3) **Decrypt** (S_k, c)
Calculate original message m by taking modulus p then modulus 2 *i.e.* $m = (c \bmod p)_2$

4) **Evaluate** $(P_k, C, c_1, \ldots\ldots, c_t)$
Binary addition and multiplication are performed on t ciphertexts using a circuit C with t input bits. Addition and multiplication gates are evaluated and the resulting integer is returned.

IV. SYSTEM MODEL

A. NETWORK MODEL

Single data owner/data user *(DO/DU)* model with Asymmetric SE is considered for ease of implementation and understanding. The single data owner/data user *(DO/DU)* is synonymous with the *Client* in the *Client Server Model* with the cloud server *CS* as the *server*. DGHV over integers [50] is used for encrypting documents/files and search queries. The scheme is designed for a serverless cloud environment such that after the initial outsourcing of encrypted data, the function is triggered only when a search query is generated. There are six phases in this system model shown in Figure 1. The *DO* generates public and private key pairs. It is assumed that the public key of the *DO* is not broadcasted and is used for encryption of files/documents and trapdoors. They can only be decrypted using the private key of the *DO*, hence providing data confidentiality. After the key pairs are generated, the *DO* encrypts his/her files/documents using his/her secret key. These files/documents go through AES encryption. The encrypted files /documents are then uploaded to the *CS*. The *CS* allows the *DO* to upload encrypted data at his/her discretion.

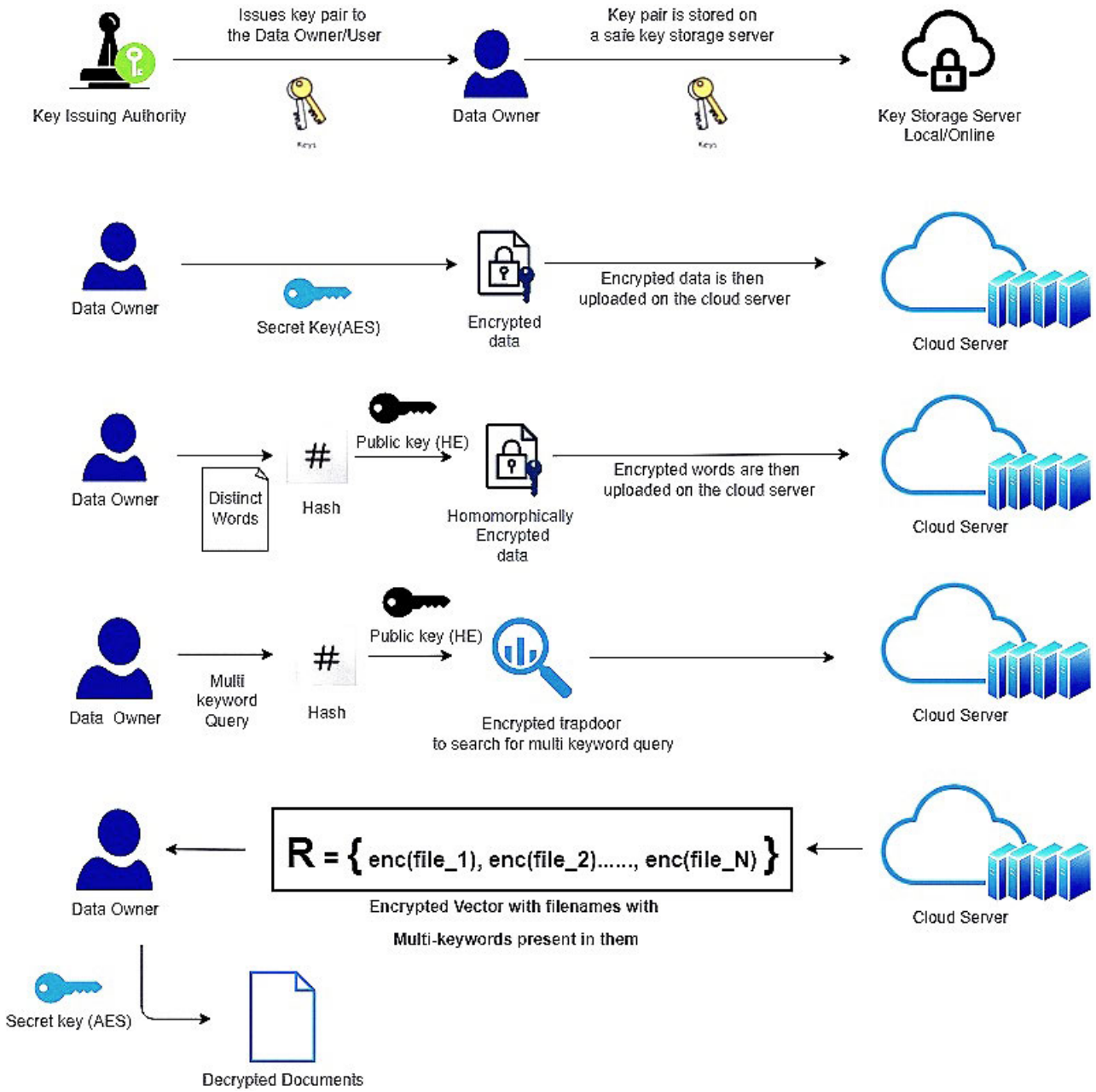

FIGURE 1. Detailed architecture.

The files/documents of *DO* are pre-processed before performing HE *i.e.* distinct words are picked and then stored in the corresponding file. The chosen words are hashed and homomorphically encrypted and then uploaded on the cloud. Once the *DO* has uploaded his files to the *CS*, it can perform SE using encrypted trapdoors. The *DO* will be able to issue a multi-keyword trapdoor to perform a search on the encrypted files stored on the *CS*. The trapdoor is encrypted and then sent to the *CS*.

When the *CS* receives the trapdoor, it then performs homomorphic operations on the stored files and trapdoor to get the encrypted resultant vector. The *CS* sends this encrypted resultant vector to the *DO*. The *DO* uses the private key to decrypt the encrypted resultant vector to get the required encrypted file names. The *DO* downloads these encrypted files and uses his secret key to decrypt them to get the search results.

B. THREAT MODEL

The threat model assumes a passive adversary, whereby the *CS* is *honest-but-curious* and uses every piece of information it can learn to make inferences, deduce results and exploit such information for its own benefit. The *CS* has trivial

information related to the data stored on the cloud, such as data size and description. The *CS* also has search queries and their corresponding documents/files [19], [51]. The *CS* can use the available information to exploit data security and obtain sensitive information.

C. MULTI-KEYWORD SE (MKSE)

MKSE algorithm consists of six phases *i.e.* Setup, Encryption, Trapdoor Generation, DWE, Multi-keyword Search, and Decryption:

1) **Setup** $(1^\lambda) \rightarrow (S_k, P_k, K_p)$ It is a probabilistic algorithm to generate a private HE key S_k, secret AES K_p, and public HE key P_k on the client end. It takes a security parameter λ as input and then outputs S_k, P_k, and K_p to enable encryption and decryption on the client side.
2) **DocEnc** $(K_p, D, N) \rightarrow C_{AES}$ Document encryption takes input K_p, set of documents D, number of documents N, and outputs corresponding encrypted documents C_{AES} . The encryption algorithm is executed at the client end.
3) **DWE** $(P_k, D, N) \rightarrow C_{HE}$ Words encryption is a probabilistic algorithm, executed at the client's end that takes input P_k, D, number of documents N, and outputs corresponding encrypted documents C_{HE}.
4) **TrapGen** $(Q, P_k) \rightarrow T$ Trapdoor Generation takes input multi-keyword query Q consisting of keywords kw, and the public key P_k to generate T. This algorithm is run at the client end to search over a set of encrypted data.
5) **MultiKS** $(T, C_{HE}, N) \rightarrow R$ Multi-keyword search algorithm takes input encrypted multi-keyword query T, set of encrypted documents C and the number of documents N to perform search and returns encrypted response R to the client. This algorithm is executed at the cloud server *CS*.
6) **Dec** $(K_p, S_k, C_{AES}, R, N) \rightarrow F$ The phase takes place at the user's end. Decryption algorithm takes the secret key (AES) K_p, private key (HE) S_k, set of encrypted documents C, encrypted vector returned to the client R, number of documents N and outputs decrypted set of documents F.

D. CORRECTNESS

The correctness of a MKSE scheme can be verified if for λ, the key pair S_k, P_k generated by Setup(1^λ), for encrypted documents C_{AES} output by DocEnc(P_k, D, N), the search using the trapdoor T returns the corresponding keywords kw present in the documents.

The correctness of the MKSE scheme can be achieved if the following holds true:

a. For $kw \in D$;

$$MultiKS(N, T, C_{HE}) = kw \cap Decrypt(K_p, C_{AES}, N, R) = kw$$

b. For $kw \notin D$;

$$MultiKS(N, T, C_{HE}) = kw \cap Decrypt(K_p, C_{AES}, N, R) = 0$$

E. SOUNDNESS

The soundness of MKSE scheme can be verified if for the security parameter λ, the public-private key pairs S_k, P_k generated by Setup(1^λ), for encrypted documents C_{AES} output by DocEnc(P_k, D, N), the search outcome based on the trapdoor T returns accurate results *i.e.* the resultant response R does not contain false positives. The soundness of the MKSE scheme can be achieved if the following holds true:

a. For $kw \in D$;

$$MultiKS(T, C_{HE}, N) = 1$$

b. For $kw \notin D$;

$$MultiKS(T, C_{HE}, N) = 0$$

V. SECURITY DEFINITIONS

The security definitions for MKSE scheme are discussed as follows:

A. KEYWORD-TRAPDOOR INDISTINGUISHABILITY

Let MKSE = (Setup, DocEnc, DWE, TrapGen, MultiKS, Decrypt) be a MKSE scheme over a set of documents D, the security parameter λ, and polynomial-time adversary $A = \{A_1, A_2, \ldots.., A_N\}$ where $N \in \mathbb{N}$, consider the following experiment,

$$\begin{aligned}
&(S_k, P_k, K_p) \leftarrow Setup(1^\lambda)\\
&C_{AES} \leftarrow DocEnc(K_p, D, N)\\
&C_{HE} \leftarrow DWE(P_k, D, N)\\
&for\ 0 < i < N:\\
&(s_A, Q) \leftarrow A(s_A, T_0, T_1, T_2 \ldots . T_N)\\
&T_i \leftarrow TrapGen(Q, P_k)\\
&C_{toss} \leftarrow \{0, 1\};\\
&(s_A, Q_0, Q_1) \leftarrow A(1^\lambda)\\
&T_{C_{toss}} \leftarrow TrapGen(Q_i, P_k)\\
&C'_{toss} \leftarrow A_{N+1}(s_A, T_{C_{toss}})\\
&T_{C'_{toss}} \leftarrow TrapGen(Q_j, P_k); j \in N\\
&if C'_{toss} = C_{toss};\ output\ 1;\\
&otherwise\ output\ 0
\end{aligned} \tag{1}$$

where state of the adversary A is represented by S_A. Keyword-Trapdoor Indistinguishability is achieved by the scheme if the following holds true:

$$Pr[Kw\_Trap_A(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda)$$

1) DESCRIPTION

The experiment comprises three phases. It begins with the challenger generating a set of encrypted documents.

- **Initial Phase:** The challenger receives a query Q from the adversary/opponent and the challenger returns the corresponding trapdoor T to the opponent A till the challenger has a set of encrypted document data.
- **Challenge Phase:** During this phase, the adversary A is asked to select and submit two queries Q_1 and Q_2 of his choice. The challenger generates corresponding trapdoors T_1 and T_2. The challenger then tosses a fair coin $C_{toss} \in \{0, 1\}$ and based on the result sends the corresponding trapdoors to the opponent A.
- **Outcome Phase:** The opponent A receives the two trapdoors T_1 and T_2. He has to correctly guess if the trapdoor belongs to query $Q \in (Q_0, Q_1)$. If the opponent is able to distinguish correctly, he wins the game. Otherwise, MKSE provides *Keyword-Trapdoor Indistinguishability.*

B. TRAPDOOR-DOCUMENT INDISTINGUISHABILITY

Let MKSE = (Setup, DocEnc, DWE, TrapGen, MultiKS, Decrypt) be a scheme over a set of N documents *i.e.* D = $\{D_1, D_2, D_3, \ldots.., D_N\}$, λ, the maximum number of keywords M, and an adversary $A = \{A_1, A_2, \ldots.., A_N\}$ where $N \in \mathbb{N}$, consider the following experiment:

$$
\begin{aligned}
&(S_k, P_k, K_p) \leftarrow Setup(1^\lambda)\\
&C_{AES} \leftarrow DocEnc(K_p, D, N)\\
&C_{HE} \leftarrow DWE(P_k, D, N)\\
&for\ 0 < i < N:\\
&(s_A, T) \leftarrow A(s_A, D_0, D_1, D_2 \ldots .D_N)\\
&D_i \leftarrow MultiKS(T, C_{HE}, N)\\
&C_{toss} \leftarrow \{0, 1\};\\
&(s_A, T_0, T_1) \leftarrow A(1^\lambda)\\
&D_{C_{toss}} \leftarrow MultiKS(T, C_{HE}, N)\\
&C'_{toss} \leftarrow A_{N+1}(s_A, D_{C_{toss}})\\
&T_{C'_{toss}} \leftarrow TrapGen(Q_j, P_k); j \in N\\
&if C'_{toss} = C_{toss};\ output\ 1;\\
&otherwise\ output\ 0
\end{aligned}
\tag{2}
$$

where S_A shows the adversary's state. Trapdoor-Document Indistinguishability is achieved by the scheme if the following holds true:

$$Pr[Trap\_Doc_A(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda)$$

1) DESCRIPTION

The experiment comprises three phases where the challenger initiates by generating the set of queries.

- **Initial Phase:** An adversary A sends the queries set Q_s to the challenger which generates trapdoors T and returns the corresponding encrypted documents till the adversary has a set of trapdoors with corresponding documents.
- **Challenge Phase:** An adversary A is asked to choose two queries *i.e.* $Q = \{Q_0, Q_1\}$ such that,
  a. Q_0 and Q_1 must be unique and distinct
  b. Q_0 and Q_1 must be present in unique documents *i.e.* D_1 and D_2
  c. Q_0 and Q_1 must be present in the documents' set.
  The challenger generates trapdoors to the corresponding queries in the set Q and returns corresponding encrypted documents D_1 and D_2.
- **Outcome Phase:** The adversary/opponent A is then asked to choose the trapdoors corresponding to the document. If the adversary A guesses correctly, he wins otherwise he loses. Hence, the probability to win the game is 50%.

C. SEARCH PATTERN PRIVACY

The proposed MKSE scheme is based on probabilistic trapdoors. Such trapdoors ensure that the scheme provides search pattern privacy. Search Pattern privacy entails whether the adversary A is able to identify whether the search is conducted with the same keyword. Since the scheme relies on probabilistic trapdoors, meaning that the trapdoor generated for a particular keyword will differ each time it is generated. This makes it difficult for an adversary to track and identify keywords that have been searched multiple times.

D. ACCESS PATTERN PRIVACY

The access pattern is preserved if the adversary/opponent is not able to identify that the trapdoor Q_1 corresponds to documents D_1 and D_3. If the adversary can not determine the corresponding documents of a unique trapdoor, access pattern privacy is preserved. MKSE scheme relies on probabilistic trapdoors, hence the encrypted trapdoors will be different for the same keyword for all queries.

VI. PROPOSED WORK

This section discusses the proposed MKSE scheme. Table 2 shows notations and abbreviations. The six phases are described below:

1) **Setup:**
   Public and private key pairs are generated at the *DO* end. Public key P_k and private key S_k generation is based on DGHV key generation phase. Pseudorandom function $f(se)$ is used in the generation of public key P_k where se is the seed. Additionally, secret key K_p is generated for standard AES encryption and decryption of documents/files. $KeyGen_{AES}()$; is the standard AES key generation function. The algorithm takes λ as input and outputs P_k, S_k, and K_p.
2) **Document Encryption:**
   Documents/files of *DO* are encrypted using secret key K_p generated in the Setup phase. Using standard AES encryption, documents/files of the *DO* are encrypted and uploaded on the *CS*. The document encryption

TABLE 2. Notations and abbreviations.

Notation	Description
P_k	public key
S_k	private key
K_p	secret key (AES)
λ	security parameter
γ	bit length of x_i's
η	bit length of private key S_k
ρ	bit length of noise r_i
r	random noise integer
τ	number of x_i's in the public key
f	pseudo-random function
se	seed
kw	keyword
D	set of documents/files
N	number of documents/files
M	number of keywords
D_i	i^{th} document
C_{AES}	set of AES encrypted documents
C_{HE}	set of DGHV encrypted documents
T	Multi-keyword Trapdoor
b	maximum number of keywords in trapdoor T or multi-keyword query Q
m	maximum number of keywords in a document/file
R	Encrypted response returned to the client based on trapdoor T
Q	multi-keyword (plaintext) query
F	Decrypted set of documents/files

Algorithm 1 Setup

Input: Security parameter λ
Output: Private key (HE) S_k, Public key (HE) P_k, Secret key (AES) K_p
begin
  choose randomly prime η-bit integer p
  choose randomly odd $q_0 \in [0, 2^{\gamma}/p)$ and let $x_0 = q_0.p$
  generate a set of integers $\mathcal{X}_i \in [0, 2^{\gamma})$ for $1 \leq i \leq \tau$
  calculate $\delta_i =< \mathcal{X}_i > +\xi_i \cdot p - r_i$ where $r_i \longleftarrow \mathbb{Z} \cap (-2^{\rho}, 2^{\rho})$ and $\xi \longleftarrow \mathbb{Z} \cap [0, 2^{\lambda+\eta}/p)$
  $K_p = KeyGen_{AES}()$;
  return $S_k = p$, $P_k = (se, x_0, \delta_1, \ldots.., \delta_\tau)$, K_p
end

algorithm takes input K_p, D, and N. It performs AES encryption and outputs C_{AES}. This algorithm is executed at the client end.

3) **Distinct Words Encryption:**
The distinct words W_i are picked from the plaintext documents and then they are hashed using the SHA3 algorithm to optimize the number of homomorphic operations performed in the search phase. The hashes are then encrypted using the DGHV encryption function. This is a probabilistic algorithm that performs fully HE on the set of documents to generate an encrypted set of documents $C_{HE}from$. It is performed at the DO end. It takes input the public key K_p to encrypt the set of documents D.

Algorithm 2 DocEnc

Input: Secret key (AES) K_p, Set of documents D, Number of documents N,
Output: AES Encrypted documents C_{AES}
for $i \leftarrow 0$ **to** N **do**
  $C_{AES_i} = Encrypt_{AES}(D_i, K_p)$;
end
Return C_{AES}

Algorithm 3 DWE

Input: Public Key P_k, Set of documents D, number of documents N
Output: Set of homomorphically encrypted documents C_{HE}
while $D_i \in D$ **do**
  input = ip.open(D_i)
  for $i \leftarrow 0$ *to* e **do**
    $temp \longleftarrow W_i$
    calculate H_i = H(temp)
    calculate $c = Encrypt_{HE}(P_k, H_i)$
    $E_h \longleftarrow$ c
  end
  $Enc_i \longleftarrow E_h$
end
$C_{HE} \longleftarrow Enc_i$ Return C_{HE}

4) **Trapdoor Generation:**
Trapdoor generation is a probabilistic algorithm to generate trapdoors for the DO. Multi-keyword query $Qi.e.Q = \{kw_1, kw_2, \ldots., kw_n\}$ provided by the DO is first hashed using SHA3-224 algorithm. DGHV fully HE is used to encrypt the hashed multi-keyword query provided by the DU. It uses the public key to encrypt the trapdoor.

Algorithm 4 TrapGen

Input: Multi-keyword Query Q, Public key P_k
Output: Encrypted Multi-keyword Query T
begin
  while $kw \in Q$ **do**
    calculate H_i = H(kw)
    calculate $c = Encrypt_{HE}(P_k, H_i)$
    $E_h \longleftarrow c$
  end
  $T_i \longleftarrow E_h$
  return T
end

5) **Multi-keyword Search:**
Using multi-keyword trapdoor T and the set of encrypted documents C_{HE}, multi-keyword search is performed. Ciphertexts T_i and C_{HE_i} are subtracted (homomorphic) from each other, and the result is saved in $resultByte$. This result is then homomorphically

added by operator overloading. The K_c contains encryption of either 0 or 1 *i.e.* if the keyword is present or not. The $flag_{kw}$ is initialized with a set of encrypted ones. The size of $flag_{kw}$ is equal to the size of the trapdoor T and for every keyword kw in the trapdoor T, the flag is set to 1 *i.e.* keyword not found. The $flag_{kw}$ is then homomorphically multiplied to K_c by operator overloading. The final result represents whether the keywords present in the multi-keyword trapdoor T are found in the set of encrypted documents C_{HE}. The resultant vector is a set of encrypted document names that contains the multi keywords provided by the DO.

Algorithm 5 MultiKS

Input: Encrypted Multi-keyword Query T, Set of encrypted documents C_{HE}, Public key P_k, number of documents N
Output: Encrypted Response R
for $i \leftarrow 0$ **to** *b; b is the number of keywords in T* **do**
  $flag_{kw} = Encrypt_{HE}(One, P_k)$;
end
Return $flag_{kw}$ **for** $i \leftarrow 0$ *to N* **do**
  for $j \leftarrow 0$ *to b* **do**
    initialize $K_c \longleftarrow Encrypt_{HE}(Zero)$
    $resultByte = Subtract_{HE}(T[j], C_{HE}[i])$;
    $K_c = add_{HE}(K_c, resultByte)$;
  end
  $flag_{kw}[i] = multiply_{HE}(K_c, flag_{kw}[i])$
  for $k \leftarrow 0$ *to b; b is the number of keywords in T* **do**
  $flags_{found} += flag_{kw}$;
  end
  $R \longleftarrow flags_{found}$
end
Return R

6) **Decryption:**
The resultant vector is then decrypted using private key (HE) S_k. The retrieved documents are then decrypted using K_p. These are the documents that encompass all the keywords included in the multi-keyword trapdoor T. The DO has the option to download and decrypt any of the individual documents/files at his/her convenience.

Algorithm 6 Decrypt

Input: Secret key(AES) K_p, Private key(HE) S_k, Set of encrypted documents C, Encrypted response R, number of documents N
Output: Decrypted set of documents F
for $i \leftarrow 0$ **to** *N; N is the number of documents in R* **do**
  $filenames \longleftarrow Decrypt_{HE}(R, S_k)$
  $F = Decrypt_{AES}(filenames, K_p)$;
end
Return F

VII. SECURITY ANALYSIS

MKSE scheme provides probabilistic trapdoors and documents preserving the privacy of the client. Some trivial information is leaked during the communication between the DO and the CS. Those leakages L_1, L_2, L_3 are described below:

- **Leakage L_1:** The leakage L_1 is related to the trapdoor generation phase. Trapdoor generated in MKSE is unique for every multi-keyword query. It is assumed that every entity *i.e.* cloud server CS, adversary/opponent A has passive access to the trapdoor generated by the data owner/user. Leakage L_1 is defined as:

$$L_1 = \left\{c = H_i + 2r + 2\sum_{i=1}^{\tau}(x_i)_{P_k}\right\}$$

  Since the trapdoor is generated with probabilistic encryption, even if the adversary gets access to the encrypted trapdoor, it cannot deduce the corresponding keyword just by looking at the encrypted trapdoor. MKSE scheme provides trapdoor privacy *i.e.* change in one bit of the plaintext (query Q) will generate a completely different ciphertext (trapdoor T).
- **Leakage L_2:** The leakage L_2 is associated with the search result returned by the cloud server. The cloud server and any adversary/opponent A have access to the search results. Leakage L_2 is defined as:

$$L_2 = \left\{R(flags_{found})\right\}$$

  $flags_{found}$ represent the encrypted vector returned by the cloud server to the data owner/user. It contains encrypted file names that had the multi-keyword trapdoor present in them. Because the $flags_{found}$ is encrypted, the adversary can collect multiple responses, and try to guess which file names correspond to which trapdoor. But even with all of the accumulated information, the adversary will not be able to guess the correct relationship between the trapdoor and the set of documents since both are generated using probabilistic encryption, hence the ciphertext is different for the same plaintext every time it is encrypted.
- **Leakage L_3:** The leakage L_3 is associated with the T and the number of keywords present in it.
- **Leakage L_4:** The leakage L_4 is associated with the size of encrypted documents/files that are uploaded on the CS. Homomorphically encrypted and standard encrypted documents/files are stored on the cloud. CS is aware of the documents/files being uploaded by the data owner DO. It also has the knowledge of storage resources consumed by the DO. Leakage L_4 is defined as:

$$L_4 = \left\{C_{HE}, C_{AES}, sizeofC_{HE}, sizeofC_{AES}\right\}$$

Considering all the leakages, the scheme provides data privacy to the DO. The leakages are insignificant in terms of the content that is leaked to an adversary. Since trapdoors and files/documents are encrypted, the information gained by the adversary is inconsequential.

TABLE 3. Comparative analysis of hashing algorithms.

Properties	SHA1	SHA2	SHA3
Output Size	160	256/224, 512/384	224/256, 384/512
Block Size	512	512/1024	1152/1088, 832/576
Rounds	80	64/80	24
Security bits	< 34	112/128, 192/256	112/128, 192/256
Security Weakness	Collisions found	Collisions and Length extension attacks	-

VIII. IMPLEMENTATION

The library used for DGHV homomorphic operations is DGHVlib-v1.1 [52], [53]. It has various implementations of DGHV such as DGHV itself [50], CMNT [54] and CNT [55]. CNT [55] is chosen for DGHV implementation as it provides public-key compression and modulus switching. MKSE scheme relies on hashing algorithms before HE to enable secure search. Because of that, multiple hashing schemes are discussed in the next section to analyze their security.

A. COMPARATIVE ANALYSIS OF HASHING ALGORITHMS

Various secure hash algorithms are compared and considered to determine the most efficient and secure hashing algorithm for the MKSE scheme. A brief comparison based on block size, output size, number of rounds, and cryptographic weaknesses is given in table 4. The SHA-3 hashing algorithm is chosen for the MKSE scheme as it is secure against collision attacks and length extension attacks, as seen in the table 4.

TABLE 4. Comparative analysis of hashing algorithms.

Properties	SHA1	SHA2	SHA3
Output Size	160	256/224, 512/384	224/256, 384/512
Block Size	512	512/1024	1152/1088, 832/576
Rounds	80	64/80	24
Security bits	< 34	112/128, 192/256	112/128, 192/256
Security Weakness	Collisions found	Collisions and Length extension attacks	-

B. DATASET DESCRIPTION

The dataset used in this research is the Switchboard-1 Telephone Speech Corpus (LDC97S62) [56] which has been collected between 1990-91 under under DARPA sponsorship. The dataset is a constituent of around 2400 spontaneous conversations with an average length of 6 minutes. Around 240 hours of recorded speech has been covered in every major dialect of English (US). It contains over 120,000 unique keywords. Over 1000 files are used for the demonstration of the Multi-keyword SE (MKSE) scheme with an average file size of 5.2 KBs.

C. SYSTEM SPECIFICATIONS

The system specifications for the public cloud server *CS* (Contabo has been used in this research) and the client side are given in Table 5.

TABLE 5. System specifications.

Features	Cloud Server (Contabo)	Client
OS	Ubuntu 20.04 LTS	Ubuntu 20.04.3 LTS
RAM	60 GB	12 GB
Processor	Intel Xeon® CPU E5-2630 v4 @2.20GHz	Intel® Core™ i5-6300U @2.40 GHz
Storage	1.6 TB SSD	251 GB HDD
Cores	10 vCPU Cores (2.20 GHz each)	4 CPU Cores

IX. PERFORMANCE ANALYSIS

The implementation of the MKSE is executed in C++ using DGHV library [52] and the results are compiled using the ''matplotlib'' library of Python 3.10 that is used for graphical representation and visualization. In this section, we further discuss the computational overhead caused by each phase. Then, we discuss the network latency in the communication overhead subsection. In the end, the comparison between storage consumed by plaintext data and encrypted data is illustrated.

A. COMPUTATIONAL OVERHEAD

The computational overhead is discussed in terms of phases of the MKSE scheme in Table 6 along with a comparison of some other SE schemes. The asymptotic notations are represented for every algorithm. The time complexity for the Setup phase is $\mathcal{O}(1)$ as it generates key pairs. The standard encryption algorithm takes the total number of documents and encrypts them using AES. The trapdoor generation algorithm depends on b representing the keywords in the query Q. The multi-keyword search algorithm takes as input the number of encrypted files N, the maximum number of distinct words in a file/document M, and the number of keywords B in the query phase. The asymptotic notation for the search algorithm is given by $\mathcal{O}(Nbm)$. The decryption algorithm depends on the total number of encrypted documents N, hence the time complexity is given by $\mathcal{O}(N)$.

1) DOCUMENT ENCRYPTION

The files/documents were encrypted via AES in approximately 6 seconds for a dataset of 1000 files as illustrated in Figure 2.

2) DISTINCT WORDS ENCRYPTION

The files are pre-processed in such a way that the unique words are picked from each file and then they are hashed using the SHA3 algorithm. These hashes are then encrypted using DGHV-HE so that we can perform a search in phase 5. The time it takes for the files to go through all of these steps is shown in Figure 3, which shows that with the increase in files, the time taken by this algorithm also increases linearly.

TABLE 6. Comparison of computational complexity.

Scheme	Technique Used	Key Generation	Index Generation	Encryption	Trapdoor Generation	Searching	Decryption
[28]	Single Keyword	$\mathcal{O}(2^{\lambda})$	$\mathcal{O}(MN+M)\|\mathcal{O}(MN)$	$\mathcal{O}(N)$	$\mathcal{O}(2H+E)$	$\mathcal{O}(MN)\|\mathcal{O}(M+1)$	$\mathcal{O}(N)$
[46]	Multi Keyword	$\mathcal{O}(2^{\lambda})$	$\mathcal{O}(MN)$	$\mathcal{O}(N)$	$\mathcal{O}((2H+E)L)$	$\mathcal{O}(MN+N^2+EM^2)$	$\mathcal{O}(N)$
[47]	Single Keyword	$\mathcal{O}(1)$	-	$\mathcal{O}(NM)$	$\mathcal{O}(1)$	$\mathcal{O}(NM)$	$\mathcal{O}(NM)$
[45]	Multi Keyword	$\mathcal{O}(2^{\lambda})$	$\mathcal{O}(MN)$	$\mathcal{O}(N)$	$\mathcal{O}(M)$	$\mathcal{O}(MN+N^2)$	$\mathcal{O}(N)$
[49]	Multi Keyword	$\mathcal{O}(2E)$	$\mathcal{O}(NM)$	$\mathcal{O}(6E+2H)$	$\mathcal{O}(3E+H+P)$	$\mathcal{O}(M+1)P$	$\mathcal{O}(6E+2H)$
[48]	Single Keyword	$\mathcal{O}(2^{\lambda})$	$\mathcal{O}(M+A)$	$\mathcal{O}(2M+3A+H)$	$\mathcal{O}(2M+3A+H)$	$\mathcal{O}(M(2A+M))$	$\mathcal{O}(2M+3A+H)$
PS	Multi Keyword	$\mathcal{O}(1)$	-	$\mathcal{O}(N+NM)$	$\mathcal{O}(B)$	$\mathcal{O}(NBM)$	$\mathcal{O}(N)$

N = total number of files, M = total number of keywords, H = Hashing function, P = pairing operations, B = keyword in query phase, L = total number of bytes per document, E = exponential function, A = Homomorphic addition operation

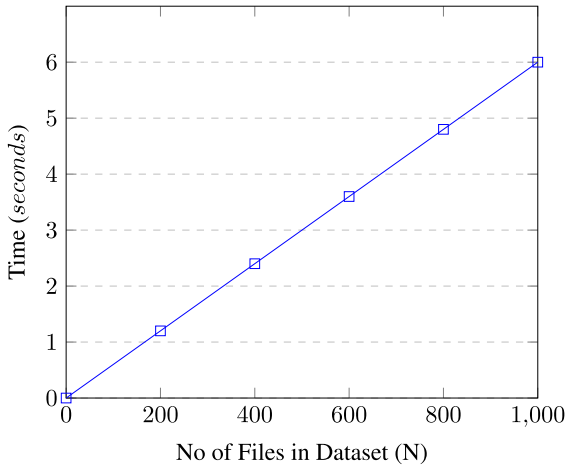

FIGURE 2. Standard encryption time.

For encrypting the 1000 files in the dataset, the DGHV encryption takes approximately 120 minutes.

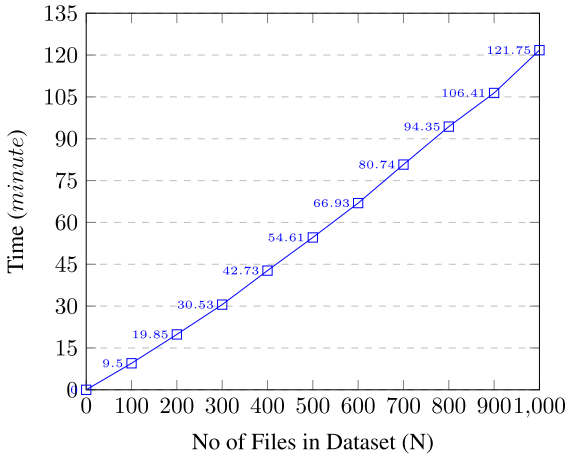

FIGURE 3. Distinct words encryption(DWE) using HE.

3) TRAPDOOR GENERATION

The time it takes to generate multi-keyword trapdoors is shown in Figure 4. The trapdoors are generated starting from two keywords to ten keywords. The graph shows that it takes 0.40 seconds to generate a 10-word trapdoor. The query in plaintext contains *'one two'* for two keywords and *'one two three'* for three keywords and so on.

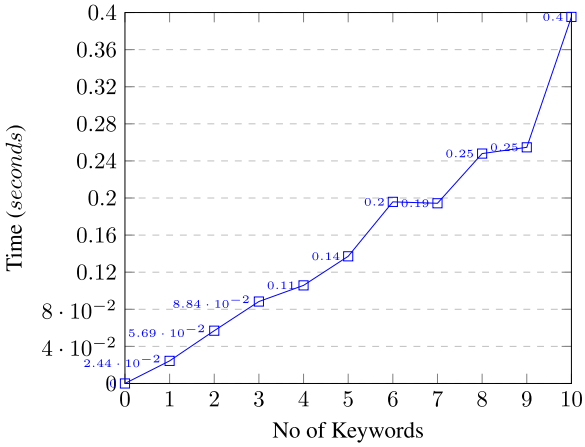

FIGURE 4. Trapdoor generation using HE.

4) MULTI-KEYWORD SEARCH

The MKSE algorithm is executed for multi-keyword trapdoors containing two, three, and four keywords as illustrated in Figure 5. The search is performed over the public cloud "Contabo". The plaintext query containing two keywords is 'hello world'. For three keywords, the plaintext query used to search on the dataset is 'I Am Here' and similarly for four keywords, the plaintext query is 'I Am Here Now'. It is worth noting that as the number of keywords to be searched for increases, the search time also increases. However, the algorithm depicts a linear growth for searching across these trapdoors.

5) DECRYPTION

The file/document decryption time is presented in Figure 6 which illustrates that the decryption algorithm takes 6 seconds to decrypt a dataset of 1000 files. The client may download a subset of these files depending on the query, which will take a negligible time.

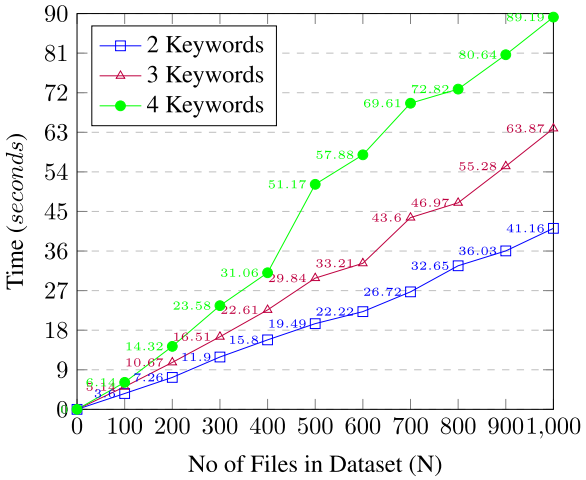

FIGURE 5. Search using homomorphic encryption.

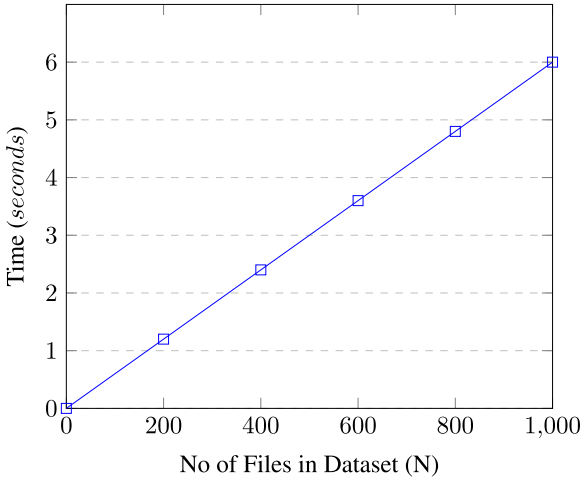

FIGURE 6. Standard decryption time.

B. COMMUNICATION OVERHEAD

This is computed by the time taken by the client to send a trapdoor to the cloud. The cloud performs the multi-keyword search and sends results to the client end. This time is illustrated in Figure 7 as a comparison between search time with latency and without latency. It may be noted that the difference is minor in both plotted lines. This shows that the latency is insignificant in comparison to the search time.

C. STORAGE OVERHEAD

This section discusses and analyzes the storage overhead of the MKSE scheme at the client and cloud server end. The storage consumption at the cloud server consists of AES encrypted files/documents (6.9 MB), and homomorphically (DWE) encrypted files/documents (24.6 GB).

At the client end, storage is consumed by DGHV public-private key pair (143.2 kB), security parameters (220 bytes), and AES secret key (64 bytes). Table 7 represents the size of the dataset before and after encryption has been performed. The increase in the dataset from MBs to GBs emphasizes the need to have a cloud platform for data storage services.

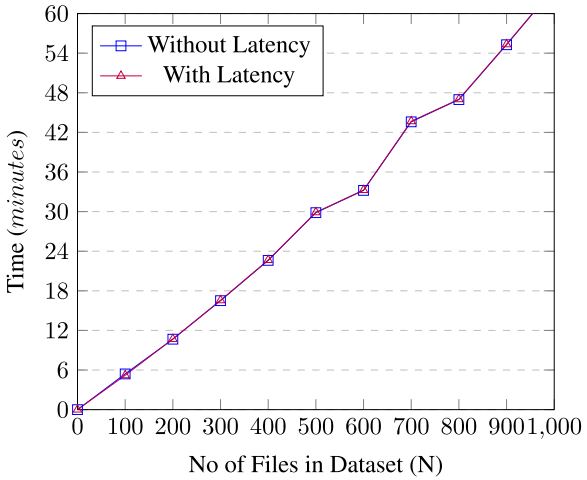

FIGURE 7. Search using HE (including latency).

TABLE 7. Data size before & after HE).

No. of Files	Unencrypted File Size (MBs)	Encrypted File Size (GBs)
100	3.149	0.73
200	6.279	1.46
300	9.185	2.10
400	11.946	2.70
500	14.059	3.11
600	16.147	3.50
700	18.221	3.89
800	20.346	4.28
900	22.478	4.68
1000	24.607	5.07

X. CONCLUSION AND FUTURE WORK

This study presents a privacy-preserving HE-based solution that enables MKSE over a serverless cloud computing domain. The security and privacy issues of the cloud data are addressed by utilizing the probabilistic encryption of DGHV. Performing search through probabilistic trapdoors on probabilistically encrypted documents has been addressed through this research which preserves the search and access patterns. The security analysis of the MKSE scheme presents possible security leakages in the proposed scheme. Further, the performance analysis of the MKSE scheme is discussed in terms of computational, communication, and storage overheads incurred during the execution of the scheme. Computational Overhead is discussed in terms of time complexity for every algorithm in the proposed MKSE scheme. Results show that the MKSE scheme is practical and provides a higher level of security in terms of privacy goals. It also gives the data owner/user the option to provide multiple words in the query to find the desired files/documents. The MKSE scheme is tested on a real-world dataset by deploying it on the public cloud "Contabo" to analyze its performance and security. The scheme provides search as well as access pattern privacy while providing the data owner/user with accurate results. In the future, we plan to introduce parallel processing thus ensuring performance enhancements.

REFERENCES

[1] L. Davis. (Sep. 2021). *What is Cloud Computing? Everything You Need to Know*. [Online]. Available: https://www.forbes.com/advisor/business/what-is-cloud-computing/

[2] C. Preimesberger. (Sep. 2021). *Benefits of Cloud Computing: The Pros and Cons*. [Online]. Available: https://www.zdnet.com/article/cloud-computing-pros-and-cons/

[3] A. Singh and K. Chatterjee, "Cloud security issues and challenges: A survey," *J. Netw. Comput. Appl.*, vol. 79, pp. 88–115, Feb. 2017.

[4] A. A. Prakash and K. S. Kumar, "Cloud serverless security and services: A survey," in *Applications of Computational Methods in Manufacturing and Product Design*. Singapore: Springer, 2022, pp. 453–462.

[5] H. B. Hassan, S. A. Barakat, and Q. I. Sarhan, "Survey on serverless computing," *J. Cloud Comput.*, vol. 10, no. 1, pp. 1–29, Jul. 2021.

[6] X. Yi, R. Paulet, and E. Bertino, "Homomorphic encryption," in *Homomorphic Encryption and Applications*. Cham, Switzerland: Springer, 2014, pp. 27–46.

[7] M. K. Patra, A. Kumari, B. Sahoo, and A. K. Turuk, "Smart healthcare system using cloud-integrated internet of medical things," in *Exploring the Convergence of Computer and Medical Science Through Cloud Healthcare*. Hershey, PA, USA: IGI Global, 2023, pp. 60–83.

[8] A. A. Khan, A. A. Wagan, A. A. Laghari, A. R. Gilal, I. A. Aziz, and B. A. Talpur, "BIoMT: A state-of-the-art consortium serverless network architecture for healthcare system using blockchain smart contracts," *IEEE Access*, vol. 10, pp. 78887–78898, 2022.

[9] D. Tomaras, M. Tsenos, and V. Kalogeraki, "Practical privacy preservation in a mobile cloud environment," in *Proc. 23rd IEEE Int. Conf. Mobile Data Manage. (MDM)*, Jun. 2022, pp. 188–197.

[10] T. Hoang, A. A. Yavuz, and J. Guajardo, "A secure searchable encryption framework for privacy-critical cloud storage services," *IEEE Trans. Services Comput.*, vol. 14, no. 6, pp. 1675–1689, Nov. 2021.

[11] IBM. (Oct. 2015). *Unlock Value of Sensitive Data Without Decryption*. [Online]. Available: https://www.ibm.com/security/digital-assets/fhe/unlock-value-of-sensitive-data-without-decryption/homomorphic-encryption-use-cases

[12] S. Kamara, T. Moataz, A. Park, and L. Qin, "A decentralized and encrypted national gun registry," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, 2021, pp. 1520–1537, doi: 10.1109/SP40001.2021.00072.

[13] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Secur. Privacy. (S&P)*, May 2000, pp. 44–55.

[14] E.-J. Goh, "Secure indexes," Cryptol. ePrint Arch., Paper 2003/216, 2003. [Online]. Available: https://eprint.iacr.org/2003/216

[15] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," *J. Comput. Secur.*, vol. 19, no. 5, pp. 895–934, Nov. 2011.

[16] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2004, pp. 506–522.

[17] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.* Berlin, Germany: Springer, 2004, pp. 31–45.

[18] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," in *Proc. Int. Conf. Inf. Commun. Secur.* Berlin, Germany: Springer, 2005, pp. 414–426.

[19] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 222–233, Jan. 2014.

[20] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," in *Proc. 8th ACM SIGSAC Symp. Inf., Comput. Commun. Secur.*, May 2013, pp. 71–82.

[21] Z. Xu, W. Kang, R. Li, K. Yow, and C.-Z. Xu, "Efficient multi-keyword ranked query on encrypted data in the cloud," in *Proc. IEEE 18th Int. Conf. Parallel Distrib. Syst.*, Dec. 2012, pp. 244–251.

[22] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–5.

[23] C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling secure and efficient ranked keyword search over outsourced cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 8, pp. 1467–1479, Aug. 2012.

[24] H. Li, D. Liu, K. Jia, and X. Lin, "Achieving authorized and ranked multi-keyword search over encrypted cloud data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 7450–7455.

[25] T. Peng, Y. Lin, X. Yao, and W. Zhang, "An efficient ranked multi-keyword search for multiple data owners over encrypted cloud data," *IEEE Access*, vol. 6, pp. 21924–21933, 2018.

[26] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. S. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 127–138, Mar. 2015.

[27] Y. Miao, W. Zheng, X. Jia, X. Liu, K.-K.-R. Choo, and R. Deng, "Ranked keyword search over encrypted cloud data through machine learning method," *IEEE Trans. Services Comput.*, vol. 16, no. 1, pp. 525–536, Jan. 2022.

[28] S. Tahir, S. Ruj, Y. Rahulamathavan, M. Rajarajan, and C. Glackin, "A new secure and lightweight searchable encryption scheme over encrypted cloud data," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 4, pp. 530–544, Oct. 2019.

[29] C. Orencik, M. Kantarcioglu, and E. Savas, "A practical and secure multi-keyword search method over encrypted cloud data," in *Proc. IEEE 6th Int. Conf. Cloud Comput.*, Jun. 2013, pp. 390–397.

[30] C. Yang, W. Zhang, J. Xu, J. Xu, and N. Yu, "A fast privacy-preserving multi-keyword search scheme on cloud data," in *Proc. Int. Conf. Cloud Service Comput.*, Nov. 2012, pp. 104–110.

[31] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 2, pp. 340–352, Feb. 2016.

[32] P. Xu, S. Tang, P. Xu, Q. Wu, H. Hu, and W. Susilo, "Practical multi-keyword and Boolean search over encrypted E-mail in cloud server," *IEEE Trans. Services Comput.*, vol. 14, no. 6, pp. 1877–1889, Nov. 2021.

[33] S. I. Ahamed, F. Rahman, E. Hoque, F. Kawsar, and T. Nakajima, "S3PR: Secure serverless search protocols for RFID," in *Proc. Int. Conf. Inf. Secur. Assurance (ISA)*, Apr. 2008, pp. 187–192.

[34] Y. Miao, J. Ma, X. Liu, F. Wei, Z. Liu, and X. A. Wang, "M2-ABKS: Attribute-based multi-keyword search over encrypted personal health records in multi-owner setting," *J. Med. Syst.*, vol. 40, no. 11, pp. 1–12, Nov. 2016.

[35] Y. Shin, J. Hur, D. Koo, and J. Yun, "Toward serverless and efficient encrypted deduplication in mobile cloud computing environments," *Secur. Commun. Netw.*, vol. 2020, pp. 1–15, Aug. 2020.

[36] J. Yu, P. Lu, Y. Zhu, G. Xue, and M. Li, "Toward secure multikeyword top-k retrieval over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, vol. 10, no. 4, pp. 239–250, Jul. 2013.

[37] Y. Wang, S.-F. Sun, J. Wang, J. K. Liu, and X. Chen, "Achieving searchable encryption scheme with search pattern hidden," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 1012–1025, Mar. 2022.

[38] X. Wang, S. Yin, H. Li, L. Teng, and S. Karim, "A modified homomorphic encryption method for multiple keywords retrieval," *Int. J. Netw. Secur.*, vol. 22, no. 6, pp. 905–910, 2020.

[39] A. Arulprakash and K. SampathKumar, "Improved encryption towards data security in serverless computing," *J. Comput. Theor. Nanosci.*, vol. 17, no. 12, pp. 5256–5260, Dec. 2020.

[40] D. N. Wu, Q. Q. Gan, and X. M. Wang, "Verifiable public key encryption with keyword search based on homomorphic encryption in multi-user setting," *IEEE Access*, vol. 6, pp. 42445–42453, 2018.

[41] R. Zhao and M. Iwaihara, "Lightweight efficient multi-keyword ranked search over encrypted cloud data using dual word embeddings," 2017, *arXiv:1708.09719*.

[42] Z. Wan and R. H. Deng, "VPSearch: Achieving verifiability for privacy-preserving multi-keyword search over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 6, pp. 1083–1095, Nov. 2018.

[43] Y. Miao, R. H. Deng, K.-K.-R. Choo, X. Liu, and H. Li, "Threshold multi-keyword search for cloud-based group data sharing," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 2146–2162, Jul. 2022.

[44] Y. Yang, X. Liu, and R. H. Deng, "Multi-user multi-keyword rank search over encrypted data in arbitrary language," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 320–334, Mar. 2020.

[45] J. Liu, B. Zhao, J. Qin, X. Zhang, and J. Ma, "Multi-keyword ranked searchable encryption with the wildcard keyword for data sharing in cloud computing," *Comput. J.*, vol. 66, no. 1, pp. 184–196, Jan. 2023.

[46] Y. Ping, W. Song, Z. Zhang, W. Wang, and B. Wang, "A multi-keyword searchable encryption scheme based on probability trapdoor over encryption cloud data," *Information*, vol. 11, no. 8, p. 394, Aug. 2020.

[47] H. Malik, S. Tahir, H. Tahir, M. Ihtasham, and F. Khan, "A homomorphic approach for security and privacy preservation of smart airports," *Future Gener. Comput. Syst.*, vol. 141, pp. 500–513, Apr. 2023.
[48] Y. Tang, Y. Chen, Y. Luo, S. Dong, and T. Li, "VR-PEKS: A verifiable and resistant to keyword guess attack public key encryption with keyword search scheme," *Appl. Sci.*, vol. 13, no. 7, p. 4166, Mar. 2023.
[49] Y. Miao, Q. Tong, R. H. Deng, K. R. Choo, X. Liu, and H. Li, "Verifiable searchable encryption framework against insider keyword-guessing attack in cloud storage," *IEEE Trans. Cloud Comput.*, vol. 10, no. 2, pp. 835–848, Apr. 2022.
[50] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2010, pp. 24–43.
[51] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-abuse attacks against searchable encryption," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 668–679.
[52] C. Gong, M. Li, L. Zhao, Z. Guo, and G. Han, "Homomorphic evaluation of the integer arithmetic operations for mobile edge computing," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–13, Nov. 2018.
[53] limengfei1187. *DGHVlib-v1.1: Implementation of Homomorphic Encryption Scheme*. Accessed: Jun. 10, 2023. [Online]. Available: https://github.com/limengfei1187/DGHVlib-v1.1
[54] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *Proc. Annu. Cryptol. Conf.* Berlin, Germany: Springer, 2011, pp. 487–504.
[55] J.-S. Coron, D. Naccache, and M. Tibouchi, "Public key compression and modulus switching for fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2012, pp. 446–464.
[56] J. J. Godfrey and E. Holliman. *Switchboard-1 Release 2—Linguistic Data Consortium*. Accessed: Jun. 10, 2023. [Online]. Available: https://catalog.ldc.upenn.edu/LDC97S62

MUSFIRAH IHTESHAM received the B.E. degree in electrical (telecommunication) engineering and the master's degree in information security from the National University of Science and Technology (NUST), Islamabad, Pakistan, in 2017 and 2020, respectively. She is currently a Research Assistant of the Project "Privacy Preserving Search over Sensitive Data Store in the Cloud" funded by the National Center for Cyber Security (NCCS), Pakistan. Her research interests include privacy-preserving techniques, cloud computing, computer networks, and cryptographic solutions.

SHAHZAIB TAHIR (Senior Member, IEEE) received the B.E. degree in software engineering from Bahria University, Islamabad, Pakistan, in 2013, the M.S. degree in information security from NUST, Islamabad, in 2015, and the Ph.D. degree in information engineering from the City, University of London, U.K., in January 2019. He is currently an Assistant Professor and an Associate HOD of the Department of Information Security, NUST. He is also the Founder and the Chief Technical Officer of CityDefend Ltd., U.K. His research interests include applied cryptography and cloud security. He has been a TPC member of many international IEEE conferences. He is an Alumni of InnovateUK CyberASAP. He is a Reviewer of many high impact journals, including IEEE Transactions on Dependable and Secure Computing, IEEE Journal of Biomedical and Health Informatics, IEEE ICC, *Future Generation Computer Systems* (Elsevier), *Cluster Computing* (Springer), Sādhanā (Springer), *Science China Information Sciences* (Springer), and *Neural Computing and Applications* (Springer).

HASAN TAHIR (Senior Member, IEEE) received the B.E. degree in software engineering from Bahria University, Islamabad, Pakistan, the M.S. degree in software engineering from the College of E&ME, NUST, and the Ph.D. degree in information security from the University of Essex, U.K. He is currently an Associate Professor and the Head of Department Information Security, School of Electrical Engineering and Computer Science (SEECS), NUST. He specializes in computer security and the IoT. He teaches courses related to applied cryptography, cyber security, information security management, cloud computing security, software engineering, and software requirements analysis and design. He actively researches applications of cryptography in one to one and group settings. His research interest includes use of physically unclonable functions for securing group of devices. He was a recipient of the University of Essex Doctoral Scholarship Award. He served as a committee member in many renowned IEEE conferences.

ANUM HASAN received the M.S. degree (Hons.) in information security from Bahria University Islamabad, Pakistan, and the M.Sc. degree from Quaid-i-Azam University, Islamabad. She is currently pursuing the Ph.D. degree in information security from the National University of Sciences and Technology (NUST), Islamabad. She is also a Lecturer with the Department of Information Security, NUST. Her research interests include cyber security and AI/ML.

AIMAN SULTAN received the B.E. degree in electrical (telecomm) engineering and the M.S. degree in information security from NUST, Islamabad, Pakistan, in 2014 and 2021, respectively, where she is currently pursuing the Ph.D. degree in information security. She is a Research Assistant with NUST. Her research interests include network security, cryptographic protocols design, and cloud security.

SAQIB SAEED received the B.Sc. degree (Hons.) in computer science from International Islamic University Islamabad, Pakistan, in 2001, the M.Sc. degree in software technology from the Stuttgart Technology University of Applied Sciences, Germany, in 2003, and the Ph.D. degree in information systems from the University of Siegen, Germany, in 2012. He is currently an Associate Professor with the Department of Computer Information Systems, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia. He is also a Certified Software Quality Engineer from the American Society of Quality. His research interests include human-centered computing, data visualization and analytics, software engineering, information systems management, and digital business transformation. He is an Associate Editor of IEEE Access and *International Journal of Public Administration in the Digital Age*, besides being a member of the advisory boards of several international journals.

OMER RANA (Senior Member, IEEE) received the B.E. degree in engineering from the Imperial College of Science, Technology and Medicine (London University), in 1989, the M.Sc. degree from the University of Southampton, in 1994, and the Ph.D. degree from the Imperial College of Science, Technology and Medicine (London University), in 1998. He is currently the Dean of the International for the Physical Sciences and Engineering College, Cardiff, U.K. He is a Senior Member of ACM and a FHEA UK.