Arabian Journal for Science and Engineering (2020) 45:10709–10729
https://doi.org/10.1007/s13369-020-04829-4

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Multi-writer Multi-reader Boolean Keyword Searchable Encryption

Dhruti Sharma[1] · **Devesh Jinwala**[2]

Received: 1 January 2020 / Accepted: 23 July 2020 / Published online: 17 August 2020

Abstract
In traditional public key searchable encryption (PKSE), a data owner (writer) utilizes data user's (reader) public key to build ciphertexts. Thus, to share D data items (W keywords per item) with R readers, a writer suffers from $O(R \cdot D \cdot W)$ computational overhead. Researchers then offer numerous schemes supporting multiple readers with optimal overhead, i.e., $O(D \cdot W)$. However, these schemes support either a single-keyword search or multi-keyword search in single-writer multi-reader (SWMR) setting where a writer could share data with the known set of readers. On the other hand, the existing multi-writer multi-reader (MWMR) PKSE offers Boolean search but suffers from $O(R \cdot D \cdot W)$ computational overhead for each writer. We observe applications desiring PKSE where writers could share data with the unknown set of readers and readers could perform search across data for Boolean query. Since the existing literature lacks such schemes, we propose a multi-writer multi-reader Boolean keyword searchable encryption (MWMR-BKSE) where the separate sets of registered readers and writers are prepared. Once registered, the writer could compute ciphertexts without knowing the potential readers and the readers could search across data at any time. The computational overhead on each writer is optimal $O(D \cdot W)$. Additionally, a deregistered client would not be able to further write or read data. With MWMR-BKSE, we offer a Boolean keyword search utilizing key policy attribute-based encryption. The security of ciphertext against chosen keyword attack is proved using full security model. With the detailed theoretical analysis and extensive simulations on real-world datasets, we demonstrate the effectiveness of MWMR-BKSE.

Keywords Multi-user searchable encryption · Multi-keyword search · Key policy attribute-based searchable encryption · Full security

1 Introduction

1.1 Background

Cloud computing offers an economical platform for data storage and easy data sharing. With cloud technology, a data owner uploads data viz. emails, personalized files (pictures/videos), healthcare records, financial information, etc. onto cloud storage server to share it with the potential data users. However, data owner and cloud service provider may not be in the same trusted domain, and hence, security and privacy of the stored data would be a critical issue. Though data encryption ensures security, it makes a simple data processing such as search across data more complex. To resolve the issue, the notion of searchable encryption (SE) has been devised where search across encrypted data is performed at server without compromising data security and privacy. The operational diagram for SE is shown in Fig. 1 where (1) The data owner (writer) writes searchable ciphertext onto storage server, (2) To search across data, the data user (reader)[1] issues a search token to the server, (3) The server with the available ciphertexts and token performs search and forwards the search result to the requesting reader. Note that the data writer computes a searchable ciphertext by encrypting the set of keywords and associating them with the encrypted payload whereas data reader computes a search token by encrypting keywords involved in the chosen search query. The payload is encrypted using any standard encryption algorithm.

More precisely, in public key searchable encryption (PKSE), a data writer encrypts a set of keywords using

✉ Dhruti Sharma
sharmadhruti77@gmail.com

[1] Information Technology Department, Sarvajanik College of Engineering and Technology, Surat, Gujarat, India

[2] Computer Engineering Department, S. V. National Institute of Technology, Surat, Gujarat, India

[1] Note: Throughout the manuscript, we use the terms—data owner or writer and data user or reader interchangeably.

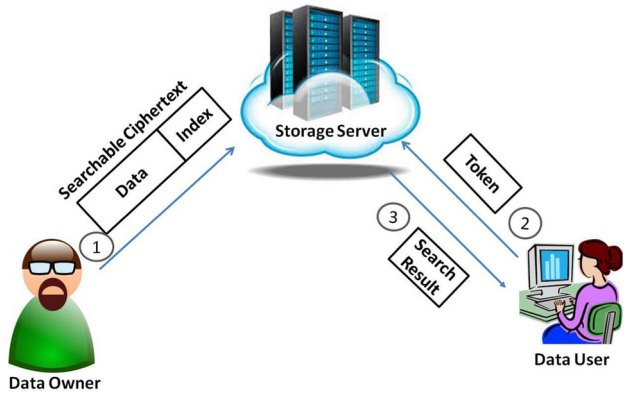


Fig. 1 Searchable encryption

data reader's public key. Consequently, a writer who wish to share D documents with W keywords per document with R readers would suffer from $O(R \cdot D \cdot W)$ computational overhead. To minimize such a computational burden on writer, numerous multi-user PKSE schemes [1–8] have been proposed. Among them, a scheme [3] offers PKSE with the reduced overhead, i.e., $O((R \cdot W) + D)$ on the writer. However, this scheme works for the static (and prefixed) group of users. In contrast, the schemes [1,2,5] support dynamic group of users where joining/leaving a group by any member is permissible even after data encryption. However, in such schemes, a writer himself is responsible for managing all the readers, and hence, a prior communication among a data writer and readers is mandatory. As a result, a writer could share data with the known set of readers only. On the other hand, the schemes [4,6–8] offer multi-reader support by employing a trusted authority (TA)-based approach. In such schemes, TA is responsible to prepare a master public key and the corresponding set of secret keys for each registered readers. A writer computes ciphertexts using a master public key and a reader computes a search token using his own secret key. Furthermore, each ciphertext serves to multiple search tokens (may be coming from different readers). Hence, these schemes support multiple readers with optimal (i.e., $O(D \cdot K)$) computational overhead on the writer. Besides, a writer could share data with the unknown set of readers. However, with the provision of a single-keyword search, these schemes are considered to be less practical. On the other hand, the TA-based schemes [9,10] with multi-keyword search functionality have been developed in the recent years that further support multiple writers and readers in the system. However, in both the schemes, TA issues a search token in response of search query from a reader, and hence, the schemes suffer from the problem of leakage of query keywords to TA. A more recent multi-writer multi-reader schemes for multi-keyword search [11,12] have resolved the keyword leakage problem by allowing each reader to compute a search token from his secret key issued by TA.

In the literature, numerous multi-keyword searchable encryption schemes offering conjunction (AND) [3,13–21] and Boolean (AND, OR) search operation [22–24] exist. However, none of these schemes support multiple readers in the system. On the other hand, though the schemes [2,9,12] support multiple keywords and multiple readers at the same time, it offers conjunctive keyword search only. Recently, the multi-user searchable encryption schemes for Boolean query have been proposed in [25–27], but these schemes support a single-writer multi-reader setting where a single data owner manages all the readers in a system and thus could share data with the known set of readers only. Furthermore, though the schemes [28,29] offer Boolean keyword search in multi-writer multi-reader setting, they incur $O(R \cdot D \cdot W)$ computational burden on each writer.

From a security perspective, in the majority of the above mentioned schemes [3,4,6–10,12–21,28], the security of ciphertexts is proved using selective security model which is considered to be a weak model. On the other hand, though the schemes [22–24] use a strong security model, i.e., full security model, they do not support multiple readers in the system. Note that the brief overview of selective and full security model is given in Sect. 2 (security perspective). To the best of our knowledge, there does not exist any fully secure PKSE scheme with the provision of Boolean keyword search in multi-writer multi-reader setting with the optimal computational overhead on data writer.

1.2 Problem Formulation

Consider an online e-health system (Fig. 2) where data from several data owners (i.e., doctors, patients, hospitals) are collected at the centralized storage server (Data repository). From a security perspective, the stored data are in the encrypted form.

Such data are supposed to be analyzed by different data users, e.g., a doctor could use patients information to provide treatment remotely, a data analyst at the research center could analyze patients record to generate periodic health analysis report, a patient could search for a doctor (specialist), an insurance company could analyze hospital data for mediclaim disbursement, etc. For each of these tasks, search across encrypted data stored on the server is indeed essential. In particular, for an online e-health system including multiple and potentially separate entities for data generation and data access, the multi-writer multi-reader PKSE with an independent set of data writers and readers is desirable. In addition, the trusted authority-based PKSE due to its

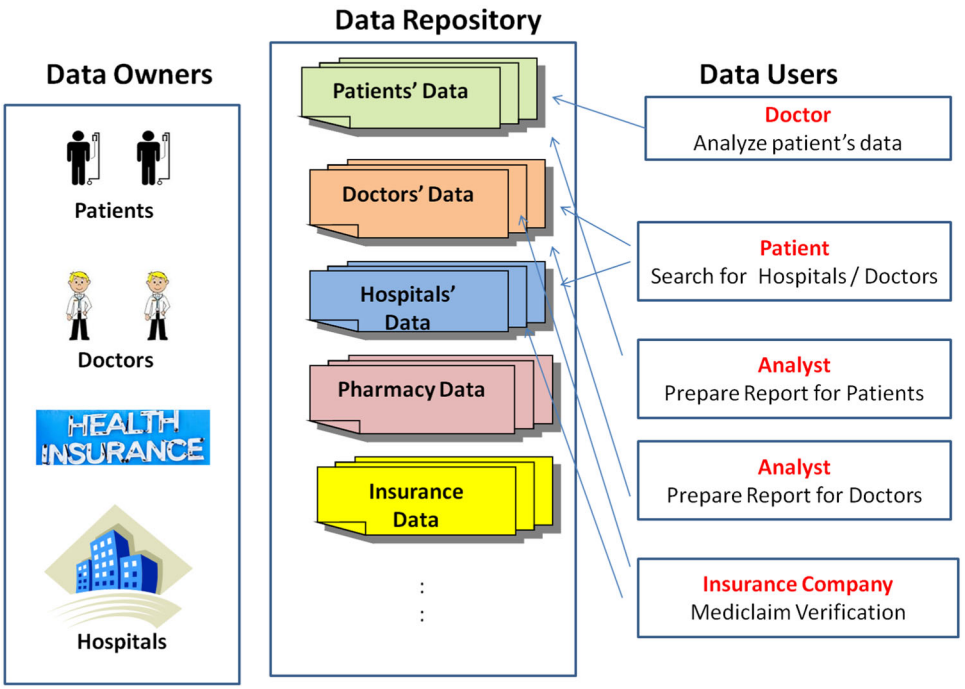


Fig. 2 Scenario of online e-health system

Table 1 Example: queries

Data user	Queries
Patient	*Find Heart Specialist in Mumbai and Ahmedabad.*
	`('Heart-Specialist') AND ('Mumbai' OR 'Ahmedabad')`
Doctor	*List Blood pressure data of patient P1 for 26-27 April 2019.*
	`('Blood-Pressure' AND 'P1') AND ('26-04-2019' OR '27-04-2019')`
Data analyst	*Find all patients taking treatment from doctor 'Dr. Desai' and having decease 'Blood Cancer' or 'Born Cancer.'*
	`('Patient' AND 'Dr. Desai') AND ('Blood Cancer' OR 'Born Cancer')`

optimal overhead on writer (as mentioned in Sect. 1.1) would be practically more effective.

Furthermore, consider a few search queries prompted by data users as mentioned in Table 1.

For each of these queries, the search is required to be performed based on Boolean relation (AND, OR) of keywords in the query. From the discussion, it could be inferred that the provision of Boolean keyword search in multi-writer multi-reader setting is indeed essential for an online e-Health system.

1.3 Our Contributions

In this paper, we propose a multi-writer multi-reader Boolean keyword searchable encryption (MWMR-BKSE). Our major contributions are listed as follows:

- **Multi-Writer Multi-Reader support** We utilize a trusted authority-based approach where a single Enterprise Trusted Authority (ETA)[2] is responsible to manage a separate set of writers and readers in the system. With such an approach, any registered writer could upload ciphertexts without prior communication with the readers and any registered reader can compute a search token from the chosen query and can ask the server to search across data. The ciphertext computational cost on the writer is independent from the number of readers, and hence, it is optimal $O(D \cdot K)$. Any new client could be a reader or writer or both but requires at least one time communication with the ETA.
- **Client Deregistration** With MWMR-BKSE, we provide deregistration algorithms executed by ETA for writers and readers. The trusted authority maintains the separate lists for registered readers and writers on the server. The

[2] We derive the term 'Enterprise Trusted Authority' based on our observation of real-world application scenarios where at least one trusted site (Enterprise central server) is always available to manage all authenticated entities viz. data writers, readers, storage server.

server before accepting a new ciphertext (from a writer) or a new token (from a reader) checks the authenticity of the writer or reader using the available lists. If authentication fails, the server rejects the ciphertext or token. Hence, a writer or a reader, once deregistered, would not be able to perform any future data storage or data access, respectively.
- **Boolean query support** We employ the notion of key policy attribute-based encryption (KPABE) [30]. In MWMR-BKSE, each keyword in a ciphertext is considered as an attribute. A search query is considered as a Boolean policy and the proposed token generation algorithm constructs a search token from such a policy.
- **Security analysis with full security model** Instead of the selective security model, we employ a full security model [31] to prove the security of the proposed scheme.

We present the detailed theoretical analysis by comparing the proposed scheme with the existing multi-user and multi-keyword searchable encryption schemes. Additionally, by performing empirical evaluation over real-world dataset, we show the effectiveness of the proposed MWMR-BKSE.

1.4 Organization of the Paper

In Sect. 2, we discuss the existing schemes related to the proposed work. The mathematical preliminaries required to devise MWMR-BKSE are given in Sect. 3. Section 4 includes the proposed scheme in terms of operational diagram, security model and algorithms. The formal construction of algorithms and security proof are presented in Sect. 5. The detailed theoretical analysis and empirical evaluation for MWMR-BKSE are discussed in Sects. 6 and 7, respectively. Finally, in Sect. 8, we put the concluding remarks.

2 Related Work

Song et al. [32] have first proposed the notion of searchable encryption where a data file is offloaded onto remote storage server as a collection of encrypted keywords. A user to perform a search for word W sends a piece of information related to W to server. The server utilizes this information to determine the encrypted documents containing W. Though the scheme [32] is practical, the search time is linear to the number of keywords to be searched since the scheme supports a single-keyword search. Moreover, the scheme [32] is not secure against statistical analysis performed with multiple queries. To resolve the problems, Goh et al. [33] and Chang et al. [34] in their separate work, construct secure searchable schemes by proposing an encrypted index for a document. Though the schemes [33,34] perform efficient search operation, they incur storage overhead proportional to the size of the index associated with each document. A searchable encryption scheme for public key setting is first proposed by Boneh et al. [35] wherein a user with his private key can search across data encrypted with the corresponding public key. However, none of the schemes [32–35] supports multi-keyword search.

Multi-keyword searchable encryption To narrow down the scope of searching and get optimal results, several searchable schemes exist with a multi-keyword search operation. In a public key setting, Park et al. [19] have proposed the first multi-keyword searchable scheme with conjunction (AND) relation among keywords. Subsequently, the schemes [3,13,15,16] provide the conjunctive searchable encryption with the improved communication and storage efficiency. On the other hand, the schemes [17,18,20] offer computationally efficient conjunctive keyword search operation. Furthermore, the search for a subset query is offered in [11,14,21]. Katz et al. [36] proposed the first disjunctive keyword (OR) search using polynomial evaluation with inner product predicate encryption. However, the computational complexity for ciphertexts in [36] is super polynomial, i.e., d^t where t is the number of variables and d is the maximum degree (of the resulting polynomial) in each variable [24]. Further, the schemes [22,23] present more efficient constructions for Boolean keyword search (AND, OR) using fully secure key policy attribute-based encryption (KPABE) mechanism. As compared to [36], the size of a ciphertext (or a token) in [22–24] is linear to the number of keywords and not the super polynomial.

Multi-user Searchable Encryption The first multi-user public key searchable encryption (PKSE) scheme is proposed by Hwang et al. [3] with storage complexity $O(D + (K \cdot R))$ for D documents with K keywords per document and R users in system. However, the search operation proposed in [3] handles conjunctive queries only. There exist several searchable encryption schemes [1,2,20,37,38] supporting groups of users where [37] handles static groups whereas [1,20,38] handle dynamic groups. Additionally, the scheme [1] offers a single-keyword search whereas the schemes [20,38] offer a conjunctive search. Recently, a multi-user scheme [2] proposes a conjunctive keyword search. However, it is practically inefficient due to its inverted-index-based construction. In addition, the scheme [2] leaks the access control information of users to the server. On the other hand, there exist several trusted authority (TA)-based multi-user searchable schemes [4,6–10,12] where TA sets up the system and prepares a master public key. In such schemes, ciphertexts are prepared by data owner (writer) using a master public key, whereas search token is computed either by TA on demand from the data user (reader) as in [9,10] or by data reader using his own search key as in [4,6–8,12]. More specifically, the schemes [9,10] are less practical due to the fact that the token generation in both the schemes is an interactive oper-

ation between TA and reader, that includes communication cost along with computational cost. Furthermore, both the schemes suffer from the problem of leakage of query keywords to TA. With the constant sized ciphertext and constant sized token, the scheme [4] provides an efficient multi-reader searchable encryption. However, in this scheme, a data owner manages all the users by issuing a secret key that further utilized in token generation by the data user. Ultimately, the scheme defines a single-writer multi-reader setting where a writer sends data to the known set of readers only. Furthermore, the ciphertext computational cost on the data owner in [4] is linear to the number of users authorized for that document, and thus, the scheme is less practical. On the other hand, the schemes [4,6–8,12] offer multi-writer multi-reader setting where a reader computes a search token utilizing his own secret key issued by the TA. Thus, any writer could share data with any number of readers. Furthermore, the scheme [8] manages queries from different authorized users by maintaining a list of authorized users (i.e., U_{List}) at the server. On the other hand, the schemes [6,7] use ciphertext policy attribute-based encryption (CPABE) to manage access control for users in system. However, among the schemes [4,6–8,12], only [12] provides multi-keyword search especially search for a conjunctive query. In addition, the scheme [27] offers a multi-user searchable encryption with Boolean query support. However in this scheme, the data owner is responsible for construction of all ciphertexts as well as management of all users, and hence, it could work in single-writer multi-reader setting. On the other hand, the recent schemes [28,29,39] offer multi-writer multi-reader setting for Boolean keyword search. However, in [28,29], the ciphertext computational overhead on writer is linear to the number of readers, i.e., $O(R \cdot D \cdot K)$. In addition, each Boolean query in [29], must include at least one conjunction operation. In [39], a reader's attribute policy is associated with each ciphertext, and hence, ciphertext computational complexity is $O(A \cdot D \cdot K)$ where A is the size of policy (in terms of access tree). A more recent scheme [11] defines a multi-keyword searchable encryption in multi-writer multi-reader setting. However, each reader in this scheme could compute search token in cooperation of the server, and thus, the overall computational cost incurred by token generation algorithm is proportional to the computational cost at the data reader as well as at the server and communication cost between reader and server. Furthermore, the scheme provides search for the subset of keywords only.

Security Perspective We observe that a majority of the existing multi-keyword searchable schemes [3,4,6–10,12–21,28] prove security of ciphertexts against chosen keyword attack using a relatively simple selective security model where an attacker is required to announce the target list of keywords before setup of public parameters. In such model, the entire keyword space is divided into two parts : (1) keywords for the computation of search tokens, and (2) keywords for the generation of challenge ciphertexts. Since the selective security model could work for the keywords prespecified for ciphertexts and token, it could be considered as a weaker form of security definition. In contrast, with a full security model, the attacker could announce the target list of keywords even after seeing the public parameters and thus could be considered as a strong form of security model. Though the existing Boolean keyword searchable schemes [22–24] prove security using a full security model, they do not support multiple readers and writers. To the best of our knowledge, there does not exist any searchable encryption scheme supporting Boolean query in multi-writer multi-reader setting with the full security model.

To fill this research gap, we offer a multi-writer multi-reader Boolean keyword searchable encryption with the optimal computational overhead on the writer along with the security analysis using a full security model. With Table 2, we show the significant features of the proposed scheme in comparison with the existing work.

3 Preliminaries

In this section, we describe mathematical preliminaries—access structure, linear secret sharing scheme (LSSS), and hardness assumptions used to define the proposed MWMR-BKSE scheme.

3.1 Access Structures

Definition 1 (*Access Structure* [40]) Let $P_1,\ldots,P_n$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $P_1,\ldots, P_n$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}\backslash\{\varnothing\}$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets. In our construction, the keywords play the role of parties.

Definition 2 (*Linear Secret Sharing Scheme (LSSS)* [23]) A secret sharing scheme $\mathcal{L}$ over a set of parties P is called linear (over Z_p) if

1. A share for each party form a vector over Z_p.
2. There exists a share generating matrix $\mathbb{A}$ with ℓ rows and m columns for $\mathcal{L}$. For all $i = 1, \ldots, \ell$, the ith row of $\mathbb{A}$ is labeled by a party $\rho(i)$ (ρ is a function from $1, \ldots, \ell$ to P). When we consider the column vector $v = (s, r_2, \ldots, r_m)$, where $s \in Z_p$ is the secret to be shared, and $r_2, \ldots, r_m \in Z_p$ are randomly chosen, then $\mathbb{A}v$ is

Table 2 Significant features of the proposed MWMR-BKSE

1Schemes	Support to			Security model	Computational overhead on writer
	Search operation	Multiple readers	Multiple writers		
Sharma et al. [9]	AND	Y	Y	Selective	$O(D \cdot K)$
Sharma et al. [12]	AND	Y	Y	Selective	$O(D \cdot K)$
Xu et al. [10]	AND	N	Y	Selective	$O(D \cdot K)$
Cui et al. [11]	Subset	Y	Y	Selective	$O(D \cdot K)$
Xu et al. [28]	AND, OR	Y	Y	Selective	$O(D \cdot K \cdot R)$
Wang et al. [27]	AND, OR	Y	N	Selective	$O(D \cdot K)$
Zeng et al. [29]	AND, OR	Y	Y	Selective	$O(D \cdot K \cdot R)$
He et al. [39]	AND, OR	Y	Y	Selective	$O(D \cdot K \cdot A)$
Han et al. [22]	AND, OR	N	N	Full	$O(D \cdot K)$
Lai et al. [23]	AND, OR	N	N	Full	$O(D \cdot K)$
Lv et al. [24]	AND, OR, NOT	N	N	Full	$O(D \cdot K)$
MWMR-BKSE	AND, OR	Y	Y	Full	$O(D \cdot K)$

D: Document to be shared, K: Number of keywords per document, R: Number of readers in System, A: Size of policy (access tree) associated with a ciphertext

the vector of ℓ shares of the secret s according to $\mathcal{L}$. The share $(\mathbb{A}v)_i$ belongs to party $\rho(i)$.

Linear Reconstruction : Assume that $\mathcal{L}$ is an LSSS for an access structure $\mathbb{A}$. Let S be an authorized set, and $I \subseteq 1, \ldots, \ell$ be defined as $I = i|\rho(i) \in S$. Then, there exist constants $\{\sigma_i \in Z_p\}_{i \in I}$, such that for any valid share $\{\lambda_i\}$ of a secret s according to $\mathcal{L}$, $\sum_{i \in I} \sigma_i \lambda_i = s$. Let $\mathbb{A}_i$ denotes the ith row of $\mathbb{A}$, we have $\sum_{i \in I} \sigma_i \lambda_i = (1, 0, \ldots, 0)$. These constants $\{\sigma_i\}$ can be found in polynomial time of the size of the share-generating matrix $\mathbb{A}$ [40]. And for unauthorized sets, no such constants $\{\sigma_i\}$ exist.

Any Boolean formula can be represented as an access tree. Any access tree with ℓ leaves can be converted in LSSS matrix of ℓ rows using the algorithm discussed in *Appendix B*.

3.2 Composite Order Bilinear Groups

Let $\mathbb{G}$ be a group generator algorithm that takes as input a security parameter 1^λ and outputs a tuple $(p_1, p_2, p_3, p_4, G, GT, e)$, where p_1, p_2, p_3, p_4 are distinct primes, G and GT are cyclic groups of order $N = p_1 p_2 p_3 p_4$. Let G_{p_1}, G_{p_2}, G_{p_3}, G_{p_4} are subgroups of order p_1, p_2, p_3, and p_4, respectively, and g is a generator of G. Let given a bilinear map $e : G \times G \rightarrow GT$ that satisfies the following properties

1. **(Bilinearity)** $\forall g, h \in G$, $a, b \in Z_N$, $e(g^a, h^b) = e(g, h)^{ab}$.
2. **(Non-degeneracy)** $\exists g \in G$ such that $e(g, g)$ has order N in GT .
3. **(Orthogonality)** $\forall h_i \in G_{p_i}$ and $h_j \in G_{p_j}$, $i \neq j$, $e(h_i, h_j) = 1$, where 1 is the identity element in GT.

3.3 Complexity Assumptions

Let the notation $s \xleftarrow{R} S$ denotes the randomly chosen element s from the set S and λ is the security parameter.

Definition 3 Given a group generator $\mathbb{G}$, we define the following distribution
$(p_1, p_2, p_3, p_4, G, GT, e) \leftarrow \mathbb{G}(1^\lambda)$, $N = p_1 p_2 p_3 p_4$, $g \xleftarrow{R} G_{p_1}, X_3 \xleftarrow{R} G_{p_3}, X_4 \xleftarrow{R} G_{p_4}, D = (G, GT, N, e, g, X_3, X_4)$, $T_1 \xleftarrow{R} G_{p_1} \times G_{p_2}, T_2 \xleftarrow{R} G_{p_1}$.
The advantage for an algorithm A in breaking the above distribution is defined as

$$Adv1_{\mathbb{G},A}(\lambda) = |Pr[A(D, T_1) = 1] - Pr[A(D, T_2) = 1]|.$$

Assumption 1 we say that $\mathbb{G}$ satisfies the distribution given in *Definition 3*, if $Adv1_{\mathbb{G},A}(\lambda)$ is a negligible function of λ for any polynomial time algorithm A.

Definition 4 Given a group generator $\mathbb{G}$, we define the following distribution
$(p_1, p_2, p_3, p_4, G, GT, e) \leftarrow \mathbb{G}(1^\lambda)$, $N = p_1 p_2 p_3 p_4$, $g, X_1 \xleftarrow{R} G_{p_1}, X_2, Y_2 \xleftarrow{R} G_{p_2}, X_3, Y_3 \xleftarrow{R} G_{p_3}, X_4 \xleftarrow{R} G_{p_4}$, $D = (G, GT, N, e, g, X_1 X_2, Y_2 Y_3, X_3, X_4)$, $T_1 \xleftarrow{R} G_{p_1} \times G_{p_2} \times G_{p_3}, T_2 \xleftarrow{R} G_{p_1} \times G_{p_3}$.
The advantage for an algorithm A in breaking the above distribution is defined as

$$Adv2_{\mathbb{G},A}(\lambda) = |Pr[A(D, T_1) = 1] - Pr[A(D, T_2) = 1]|.$$

Assumption 2 we say that $\mathbb{G}$ satisfies the distribution given in *Definition 4*, if $Adv2_{\mathbb{G},A}(\lambda)$ is a negligible function of λ for any polynomial time algorithm A.

Definition 5 Given a group generator $\mathbb{G}$, we define the following distribution $(p_1, p_2, p_3, p_4, G, GT, e) \leftarrow \mathbb{G}(1^\lambda)$, $N = p_1p_2p_3p_4$, $s \overset{R}{\leftarrow} Z_n$, $g, u, u' \overset{R}{\leftarrow} G_{p_1}$, $g_2, X_2 \overset{R}{\leftarrow} G_{p_2}$, $X_3 \overset{R}{\leftarrow} G_{p_3}$, $X_4, h, h' \overset{R}{\leftarrow} G_{p_4}$, $B_{24}, D_{24} \overset{R}{\leftarrow} G_{p_2} \times G_{p_4}$, $D = (G, GT, N, e, g, g_2, X_2, uh', u'h', g^s B_{24}, X_3, X_4)$, $T_1 \overset{R}{\leftarrow} (u^s D_{24}, u'^s D24)$, $T_2 \overset{R}{\leftarrow} G_{p_1} \times G_{p_2} \times G_{p_4}$.
The advantage for an algorithm A in breaking the above distribution is defined as

$$Adv3_{\mathbb{G},A}(\lambda) = |Pr[A(D, T_1) = 1] - Pr[A(D, T_2) = 1]|.$$

Assumption 3 we say that $\mathbb{G}$ satisfies the distribution given in *Definition 5*, if $Adv3_{\mathbb{G},A}(\lambda)$ is a negligible function of λ for any polynomial time algorithm A.

3.4 Reviewing the Existing Fully Secure Boolean Keyword Searchable Encryption Schemes

In the literature, we identify three Boolean searchable encryption schemes [22–24] that offer semantic security to ciphertexts against chosen keyword attack using full security model. Contrast to the selective security that is often provided by the typical searchable encryption schemes, full security is more practical. The prime reason behind this practicality is that with the full security model one can prove security against an attacker which might be more powerful than the attacker considered by the selective security model. More precisely, during security games in the full security model, an attacker could announce the target list of keywords even after seeing public parameters, and hence, there is no restriction on selection of keywords to perform a chosen keyword attack.

Reviewing the schemes [22–24], we observed that they provide Boolean keyword search functionality using key policy attribute-based encryption (KPABE). However, the following limitations make these schemes less adaptable in real-world application scenarios.

1. No discussion about who is the generator of public-private keys?
2. No multi-reader support. To serve multiple readers, the data owner must have to compute separate ciphertext for each reader using that reader's public key. Apparently, computational overhead on a writer and storage overhead on the server would raise to $O(R \cdot D \cdot W)$ for D documents with W keywords per document and R readers in the system.
3. No multi-writer support. Any data owner who knows the public parameter could write ciphertexts. Hence, there is no control on the writer.

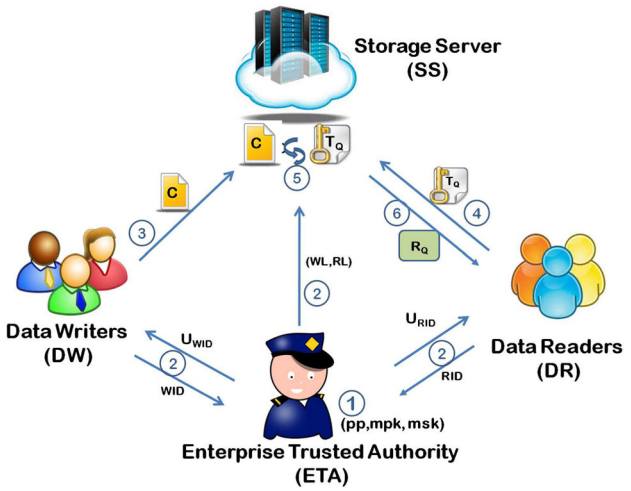


Fig. 3 Operational diagram of MWMR-BKSE

We overcome these limitations by proposing a different system model that includes an additional trusted authority besides data owner(Writer), data user(reader) and storage server. We follow the notion of KPABE-based searchable encryption offered by [22–24]. However, the prime objective behind the proposed work is to offer a more practical Boolean keyword search for a system having multiple writers and multiple readers.

4 The Proposed Scheme

In this section, we present operational diagram, algorithms and security model for the proposed MWMR-BKSE. The notations are listed in Table 3.

4.1 Operational Diagram

The operational diagram of MWMR-BKSE consists of four entities: (i) Data Writer (DW), (ii) Data Reader (DR), (iii) Storage Server (SS), (iv) Enterprise Trusted Authority (ETA) (Fig. 3).

As shown in Fig. 3, (1) Initially, ETA sets up the system by generating (pp,mpk,msk) and publishing (pp, mpk). Additionally, ETA prepares two empty lists (RL, WL) and stores them on SS, (2) During registration, ETA issues U_{RID} to each new reader (RID) and U_{WID} to each new writer (WID). For each new reader or writer, ETA updates the corresponding RL or WL on SS. Note that the registration of a new reader or writer is allowed at any time after system setup, (3) A registered DW computes a searchable ciphertext (C) by associating a list of keywords $W = \{w_1, w_2, \ldots, w_n\}$ with encrypted data M'. Here, each $w_i \in W$ is encrypted using the proposed *Encryption()* algorithm. He then sends C to SS, (4) A registered DR constructs a token T_Q from the chosen

Table 3 List of notations

Symbols	Description
λ	Security parameter
$\mathcal{KS}$	Keyword space
$\mathcal{MS}$	Message space
pp	System's public parameters
(mpk, msk)	Master public-secret key pair
(WID, RID)	A unique identification number for writer and reader
(U_{WID}, U_{RID})	A secret key for writer and reader
(S_{WID}, S_{RID})	A server side key for writer and reader
(WL, RL)	List of registered writers and readers
n	Number of keywords in ciphertext
ℓ	Number of keywords in query
W=$\{w_1, w_2, \ldots, w_n\}$	A list of n keywords
$M' = E_{key}(M)$	A payload message encrypted with symmetric cipher E with key key
C	A ciphertext
ℓ	Number of keywords in query and $\ell \leq n$
Q=$(\mathbb{A}, \rho, \mathbb{T})$	A Boolean query in terms of share generating matrix $\mathbb{A}$,mapping function ρ, set of keyword values $\mathbb{T}$
T_Q	Token for query Q
R_Q	Search Result for query Q

Boolean query Q and issues it to SS, (5) SS applies T_Q on available C by executing the proposed *Search()* algorithm, (6) SS returns a search result R_Q to DR where $R_Q = M'$ for successful search, otherwise $R_Q = \perp$.

Assumptions: (i) SS is a semi-honest server, i.e., it follows the protocol but curious to know keywords (in plaintext) from the list of encrypted keywords, (ii) All communication channels are secure, (iii) The payload $M' = E_{key}(M)$ where E is any symmetric encryption cipher with a key, (iv) ETA assigns a unique identification number RID (resp. WID) to each data reader (resp. writer) based on the physical document verification.

4.2 Security Model

We claim that the proposed MWMR-BKSE is fully secure with indistinguishability-based security against chosen keyword attack (IND-CKA) performed by an active attacker $\mathcal{A}$ who is able to obtain a token T_Q for any Boolean query Q of his choice. This implies that with MWMR-BKSE, an attacker $\mathcal{A}$ can't distinguish a ciphertext C_{W_i} with the list of keywords W_i from a ciphertext C_{W_j} with the list of keywords W_j unless having the corresponding tokens. The formal definition of security for the proposed approach is given by the following game [23,24] between an active attacker $\mathcal{A}$ and a challenger $\mathcal{C}$.

- **Setup** The challenger $\mathcal{C}$ runs the *Setup()* algorithm and provides a master public key mpk to an attacker $\mathcal{A}$.
- **Phase 1** $\mathcal{A}$ adaptively asks for some tokens $T_{Q_{W'}}$ of a query $Q_{W'}$ to $\mathcal{C}$.
- **Challenge** At some point in between, $\mathcal{A}$ asks $\mathcal{C}$ for *Encryption()* using a set of keywords W_0 and W_1 as a challenge for which he doesn't know any token T_{W_i}. In response, $\mathcal{C}$ flips a random coin $b \in \{0, 1\}$ and gives a challenge ciphertext C_{W_b} to $\mathcal{A}$.
- **Phase 2** $\mathcal{A}$ asks for token $T_{Q_{W''}}$ for another query $Q_{W''}$ where $W'' \neq \{W_0, W_1\}$.
- **Guess** $\mathcal{A}$ outputs $b' \in \{0, 1\}$ and wins the game if $b' = b$.

The advantage to break the system for $\mathcal{A}$ is as follows.

$$Adv_{\mathcal{A}}(\lambda) = |Pr[b' = b] - \tfrac{1}{2}|.$$

We say that the proposed MWMR-BKSE is IND-CKA secure if $Adv_{\mathcal{A}}(\lambda)$ is negligible for any polynomial time attacker $\mathcal{A}$.

4.3 Algorithms

The algorithms defined for MWMR-BKSE are as follows:

Algorithm 1: Setup

With *Setup* algorithm, ETA initializes the system parameters.
Input: λ, $\mathcal{MS}$
Output: pp, mpk, msk, n

1 Defines the keyword space $\mathcal{KS}$ of n keyword fields from input message space $\mathcal{MS}$.
2 Generates public parameter pp and a master public - secret key pair (mpk, msk).
3 Prepares lists (RL, WL) for maintaining the registered readers and writers, respectively. It then stores $(RL = NULL, WL = NULL)$ onto SS.

Algorithm 2: W_Registration

With *Writer Registration* algorithm, ETA registers a new writer possessing WID.
Input: pp, mpk, WID
Output: (U_{WID}, S_{WID}), Updated WL

1 For a writer possessing WID, it computes a secret key U_{WID}.
2 From (mpk, U_{WID}), it computes the server side key S_{WID}.
3 Appends (WID, U_{WID}) into WL.
4 Replaces WL at server with the modified WL computed in step 3.

Algorithm 3: R_Registration

With *Reader Registration* algorithm, ETA registers a new reader possessing RID.
Input: pp, msk, RID
Output: (U_{RID}, S_{RID}), Updated RL

1 From msk, computes a secret key U_{RID} for writer.
2 From (msk, U_{RID}), computes the server side key S_{RID} for writer.
3 Appends (RID, U_{RID}) into RL.
4 Replaces RL at server with the modified RL computed in step 3.

Algorithm 4: Encryption

With *Encryption* algorithm, DW possessing WID computes a ciphertext C.
Input: pp, mpk, U_{WID}, W, M'
Output: Ciphertext C
1 Using (pp, mpk, U_{WID}), it encrypts each keyword w_i in list $W = \{w_1, w_2, \ldots, w_n\}$.
2 Using (pp, mpk), it computes different ciphertext components.
3 Generates a ciphertext C from the input encrypted message M' and encrypted keywords as well ciphertext components computed in step 1 and 2, respectively.

Algorithm 5: C_Accept

SS executes the *Ciphertext Accept* algorithm to check the authenticity of the writer sending the ciphertext C.
Input: C, WL, S_{WID}
Output: -
1 Checks the existence of $WID \in C$ in the available WL.
2 If C comes from the genuine writer, it computes C' from (C, S_{WID}) and stores C' in the storage area, Otherwise it rejects C.

Algorithm 6: TokGen

With *Token Generation* algorithm, DR computes a search token.
Input: pp, mpk, U_{RID}, Q
Output: T_Q
1 USing (pp, mpk, U_{RID}), it computes a token T_Q from the input Boolean query Q=($\mathbb{A}$, ρ, $\mathbb{T}$). Note that the detailed description of Q is discussed in *Section 5.1*. DR then sends this token to the server SS.

Algorithm 7: Search

With this algorithm, SS performs search across the available ciphertexts C.
Input: pp, U_{RID}, Q, S_{RID}
Output: T_Q
1 Using available RL, it first checks the authenticity of the received T_Q.
2 If T_Q comes from the genuine reader, it performs Boolean search by applying T_Q on the available C using S_{RID}.
3 If search is successful, the algorithm returns $R_Q = M'$. Otherwise, it returns $R_Q = \perp$.

Algorithm 8: W_Deregistration

With *Writer Deregistration* algorithm, ETA revokes the registered writer.
Input: WID, WL
Output: -
1 Update local copy of WL by removing the entry of a registered writer possessing WID.
2 Replaces WL at server with the modified WL computed in step 1.

Algorithm 9: R_Deregistration

With *Reader Deregistration* algorithm, ETA revokes the registered reader.
Input: RID, RL
Output: -
1 Update local copy of RL by removing the entry of a registered reader possessing RID.
2 Replaces RL at server with the modified RL computed in step 1.

5 Multi-writer Multi-reader Boolean Keyword Searchable Encryption (MWMR-BKSE)

In this section, we elaborate the proposed algorithms with the formal constructions. The security proof using full security model is also discussed.

5.1 Construction

In MWMR-BKSE, each query Q is considered as a policy (Boolean formula). Formally, a Boolean formula is represented as an LSSS access structure, i.e., Q=($\mathbb{A}$, ρ, $\mathbb{T}$). Here, $\mathbb{A}$ is an $\ell \times m$ share generating matrix where the number of rows, i.e., ℓ is equal to the number of keywords in Q. We assume that no keyword is repeated in query Q. ρ is a mapping function that maps each row of $\mathbb{A}$ to a single keyword of n available keywords (i.e., $\rho : \{1, \ldots, \ell\} \rightarrow \{1, \ldots, n\}$). $\mathbb{T}$ is a set of values of keywords $(t_{\rho(1)}, t_{\rho(2)}, \ldots, t_{\rho(l)})$ specified by a Boolean query Q.

In such a setup any input ciphertext with a set W, satisfies a query Q if and only if there exists a subset $I \subseteq \{1, 2, \ldots, \ell\}$ and a constant $\{\sigma_j\}$ such that

$$\sum_{j \in I} (\sigma_j \mathbb{A}_j) = (1, 0, \ldots, 0) \quad and \quad w_j = t_{\rho(j)}, \forall j \in I. \quad (1)$$

Here, $\mathbb{A}_j$ is the jth row of matrix $\mathbb{A}$. Similar to I, multiple subsets can be defined that satisfy the Boolean query Q.

The concrete constructions for the proposed algorithms are as follows:

5.2 Security Analysis

In the proposed construction, the components of ciphertext C includes elements from subgroups G_{p_1} and G_{p_4}. On the other hand, the components of token T include elements from subgroups G_{p_1} and G_{p_3}. Thus, ciphertext C_W does not reveal any information about plaintext keywords unless token T_Q for query Q containing keywords $W' = W$ is available. We define *Theorem 1* for proving the security of the proposed scheme.

1. Setup(λ, $\mathcal{MS}$)

1 From input message space $\mathcal{MS}$, the algorithm identifies the potential keywords and define a keyword space $\mathcal{KS}$ containing n keyword fields.
2 Runs a generator algorithm $\mathbb{G}$ to obtain (p_1, p_2, p_3, p_4, G, GT, e). Here $G = G_{p_1} \times G_{p_2} \times G_{p_3} \times G_{p_4}$, G and GT are cyclic groups of order $N = p_1 p_2 p_3 p_4$. The algorithm then computes a master public key mpk and a master secret key msk as follows:
   – Selects $\alpha, \beta_1, \beta_2, \ldots, \beta_n \in Z_N$, $g, u, u_1, u_2, \ldots, u_n \in G_{p_1}$, $X_3 \in G_{p_3}$, $X_4, h_1, h_2, \ldots, h_n \in G_{p_4}$ at random.
   – Computes $H_i = (u_i \cdot h_i)$, $u_i' = u^{\beta_i}$ for $1 \le i \le n$.
   – Sets pp=($G, GT, e, g, n, \mathcal{KS}$), mpk=($N, g, g^{\alpha}, H_i, X_4$) and msk=(α, u_i', X_3) for $1 \le i \le n$.
3 Sets up two empty lists $RL = \{\perp, \perp\}$ and $WL = \{\perp, \perp\}$ and stores them onto SS.

2. W_Registration(pp, mpk, WID)

1 Selects $x_{WID} \in Z_N$ at random and sets a secret key $U_{WID} = x_{WID}$.
2 From (mpk, U_{WID}), it computes $y_{WID} = g^{-x_{WID}}$ and sets server side key $S_{WID} = (WID, y_{WID})$.
3 Updates the local copy of WL as $WL = (WL) \cup \{WID, S_{WID}\}$.
4 Replaces WL stored at SS with the modified WL.

3. R_Registration(pp, mpk, msk, RID)

1 Selects $x_{RID}, x'_{RID} \in Z_N$ at random.
2 Computes $y_{RID} = \alpha / x_{RID}$ and $u'_{i\_RID} = (u'_i)^{1/x'_{RID}}$ for $1 \le i \le n$.
3 Sets a read secret key $U_{RID} = (x_{RID}, u_i, u'_{i\_RID})$ and a server side key as $S_{RID} = (RID, y_{RID}, x'_{RID})$.
4 Updates the local copy of RL as $RL = (RL) \cup \{RID, S_{RID}\}$.
5 Replaces RL stored at SS with the modified RL.

4. Encryption(mpk, W, U_{WID}, M')

1 Selects $s \in Z_N$, h, Z_0, Z_{1i}, $Z'_{1i} \in G_{p_4}$ at random.
2 From input mpk and $W = \{w_1, w_2, \ldots, w_n\}$, it computes ciphertext components CC =(C_0, C'_0, C_i, C'_i) as follows
$C_0 = e(g, g^{\alpha})^s$, $C'_0 = (gh)^s \cdot Z_0$, $C'_i = (u'_i)^s \cdot Z'_{1i}$, $C_i = ((H_i)^{w_i})^s \cdot Z_{1i} \cdot g^{U_{WID}}$ for $1 \le i \le n$.
3 Sets a ciphertext $C = (CC, WID, M')$ and sends it to SS.

5. C_Accept(C, WL)

1 For input $WID \in C$, the algorithm checks WL.
2 If WID is found in WL then the algorithm
   – Replaces C_i=$C_i \cdot y_{WID}$=$((H_i)^{w_i})^s \cdot Z_{1i}$
   – Stores C on storage space.
3 Else
   – Rejects input ciphertext C.

6. TokGen(pp, mpk, U_{RID}, Q)

1 For the input Boolean query $Q = (\mathbb{A}, \rho, \mathbb{T})$, it first selects a random vector v such that $\mathbb{A}_i \cdot v = x_{RID}$ for $1 \le i \le \ell$.
2 It selects $r_i \in Z_N$, $V_{0i}, V_{1i}, V_{2i}, V_{3i} \in G_{p_3}$ at random for $1 \le i \le \ell$.
3 Then computes $D_{0i} = g^{\mathbb{A}_i v} \cdot V_{0i}$, $D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot V_{1i}$, $D_{2i} = (u'_{i\_RID})^{r_i} \cdot V_{2i}$, $D_{3i} = g^{r_i} \cdot V_{3i}$.
4 From ($\mathbb{A}$, ρ), DR computes a set of minimum subset I' (that can satisfy a query Q) as $I' = \{(I_1, \sigma_1), (I_2, \sigma_2), \ldots\}$ where $I_k \subseteq \{1, 2, \ldots, \ell\}$ and $\sigma_k = \{\sigma_{jk} | 1 \le j \le |I_k|\}$ satisfying Eq. (1).
5 Finally, DR possessing RID outputs a token for query Q as $T_Q = (D_{0i}, D_{1i}, D_{2i}, D_{3i}, I', RID)$ for $1 \le i \le \ell$.

7. Search($pp, mpk, C, T_Q, S_{RID}, RL$)

1 For input $RID \in T_Q$, it checks RL.
2 If RID is found in RL then algorithm
   – Computes $D'_{1i} = (D_{0i})^{y_{RID}} \cdot D_{1i} = (g^{\mathbb{A}_i v}) \cdot V_{1i} \cdot (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot V_{2i}$, $D'_{2i} = (D_{2i})^{x'_{RID}} = (u'_i)^{r_i} \cdot V_{3i}^{x'_{RID}}$
   – Performs search as
     For each $I_k \in I'$
       For each $j \in I_k$
         Checks

$$C_0 =^{?} \prod \left(\left(\frac{e(C'_0, D'_{1j})}{e(C_j, D_{3j})} \cdot \frac{e(C'_0, D'_{2j})}{e(C'_j, D_{3j})} \right)^{\sigma_{jk}} \right) \tag{2}$$

   – If check in Eq. (2) is successful, then algorithm sends encrypted data associated with this ciphertext as result, i.e., $R_Q = M'$ to reader RID, otherwise, it returns $R_Q = \perp$.
3 Else
   – Rejects input token T_Q.

8. W_Deregistration(WID, WL)

1 It revokes a writer WID by updating local copy of WL as WL = WL - $\{WID, S_{WID}\}$.
2 Replaces WL at server with the modified WL.

9. R_Deregistration(RID, RL)

1 It revokes a writer RID by updating local copy of RL as RL = RL - $\{RID, S_{RID}\}$.
2 Replaces RL at server with the modified RL.

Theorem 1 *The proposed MWMR-BKSE is fully secure, if assumptions 1, 2, and 3 holds.*

Proof Following the approach of [24], we define two structures : semi-functional ciphertext and semi-functional keys. These structures are used only in the security proofs and not in the real system. The proof consists of three steps:

1. *Construction of semi-functional ciphertexts and semi-functional keys. Semi-functional Ciphertext*: Let g_2 is a generator of G_{p_2}. A semi-functional ciphertext is constructed as follows:

– Using *Encryption()* algorithm, a normal ciphertext $C = (C_0, C_0', C_i, C_i')_{1\le i\le n}$ is prepared.
– Then, for each (C_i, C_i') components, the random values $\gamma_i, \gamma_i' \in Z_N$ are chosen.
– For any random $c \in Z_N$, a semi-functional ciphertext is prepared as

$$C' = (C_0, \quad C_0' \cdot g_2^{c}, \quad C_i \cdot {g_2}^{c\gamma_i}, \quad C_i' \cdot {g_2}^{c\gamma_i'})_{1\le i\le n}.$$

Semi-functional Keys: A semi-functional key will be of two types : Type 1 or Type 2. A semi-functional key can be constructed as follows:

– Using *TokGen()* algorithm, a normal token key $T_Q = (D_{0i}, D_{1i}, D_{2i}, D_{3i}, I', UID)_{1\le i\le \ell}$ is generated for a Boolean query $Q = (\mathbb{A}, \rho, \mathbb{T})$.
– A random vector $z \in Z_N^m$ is chosen and $\delta_i = \mathbb{A}_i \cdot z$ is computed.
– A random value $\eta_i \in Z_N$ is chosen.
– Type 1 semi-functional token key $T_{1Q}' = (D_{0i}', D_{1i}', D_{2i}', D_{3i}', I', UID)$ is prepared as $D_{0i}' = D_{0i} \cdot {g_2}^{\delta_i}$, $D_{1i}' = D_{1i} \cdot {g_2}^{\eta_i \gamma_{\rho(i)}}$, $D_{2i}' = D_{2i} \cdot {g_2}^{\eta_i \gamma_{\rho(i)}'}$, $D_{3i}' = D_{3i} \cdot {g_2}^{\eta_i}$
– The Type 2 semi-functional key $T_{2Q}' = (D_{0i}', D_{1i}', D_{2i}', D_{3i}', I', UID)_{1\le i\le \ell}$ is computed without the terms ${g_2}^{(\eta_i \gamma_\rho(i))}$ and ${g_2}^{(\eta_i \gamma_\rho(i)')}$ as $D_{0i}' = D_{0i} \cdot {g_2}^{\delta_i}$, $D_{1i}' = D_{1i}$, $D_{2i}' = D_{2i}$, $D_{3i}' = D_{3i}$.

2. *Defining a sequence of attack games.*
We give security proof by defining the attack games. The first game is $Game_{real}$. It is a real security game already defined in Sect. 4.3. The next game is $Game_0$ in which all token keys are normal, but challenge ciphertexts are semi-functional. Let q is the maximum number of token key queries and, k is the actual number of key queries made by an attacker.
For $1 \le k \le q$ and $1 \le \zeta \le n$, we define

– $Game_{k,1}$: In this game, a challenge ciphertext is semi-functional, the first $k-1$ token keys are semi-functional of Type 2, the kth token key is semi-functional of Type 1, and the remaining token keys are normal.
– $Game_{k,2}$: In this game, a challenge ciphertext is semi-functional. The first k token keys are semi-functional of Type 2, and the remaining token keys are normal.
– $Game_{Final_\zeta}$ or $Game_{q,2}$: In this game, all token keys are semi-functional of Type 2, and the challenge ciphertext $C_b = (C_0, C_0', C_i, C_i')_{1\le i\le n}$ is a semi-functional encryption of W_b where the elements $C_1, \ldots, C_\zeta$ and $C_1', \ldots, C_\zeta'$, are randomly chosen from $G_{p_1} \times G_{p_2} \times G_{p_4}$.

3. *Proving these attack games are indistinguishable with the real game.* The proof of Indistinguishability of above games is given by four lemmas described in *Appendix A*. It is clear that in $Game_{Final_\zeta}$, the attacker's advantage is negligible since the challenge ciphertext C_b is independent of the keyword sets W_0 and W_1. Thus, we conclude that the advantage of an adversary in $Game_{real}$ is negligible. This completes the proof of *Theorem 1*. □

6 Theoretical Analysis

In this section, we carry out the comparative analysis of MWMR-BKSE with the existing multi-keyword searchable encryption schemes [9,11,12,22–24,27–29,39].

6.1 Storage Overhead

The storage overhead is presented in terms of ciphertext size (CS) (excluding the size of payload data) and token size (TS) (Table 4). From table, we observe that CS for the proposed MWMR-BKSE is linear to the number of keywords in ciphertexts, i.e., n as that in the other listed schemes. More precisely, with $((1+2n)G + G_T)$ ciphertext size, the proposed scheme offers moderate ciphertext storage overhead which is lower than [11,27–29,39] and higher than [9,12,22–24]. Such overhead is due to the encryption of keywords in ciphertexts using DW's secret key and not using master public key. On the other hand, with constant TS, the schemes [9,12] offer optimal token storage overhead. However, they have provision for conjunctive(AND) keyword search. In contrast, though the scheme [27] provides Boolean (AND, OR) keyword search, it does not support multiple writers in the system (as shown in Table 2). In addition, the schemes [9,12,27] use a relatively weak, selective security model (Table 2) to prove the security of ciphertext. We additionally remark that though the schemes [22–24] provide full security for ciphertext with TS considerably less than MWMR-BKSE, they do not have provision for the multi-writer multi-reader setting (Table 2). Furthermore, we note that though TS in [39] seems lower than the proposed scheme, with only one attribute in attribute policy (could be impractical)[3] associated with token, the token size becomes exactly the same as the TS in MWMR-BKSE.

[3] In the field of security, controlled data access based on a single component (i.e., identity or attribute) could be offered by identity-based encryption (IBE). However, to control access of data using multiple attributes of a user is the primary focus of attribute-based encryption (ABE).

Table 4 Comparative analysis: storage-computational complexity

Schemes	Storage overhead		Computational overhead		
	Ciphertext size	Token size	Encryption()	TokGen()	Search()
Sharma et al. [9]	$(1+n)G^*$	$(3)G^*$	$(2+2n)M$	$(6)M$	$2M+2P$
Sharma et al. [12]	$(1+n)G*$	$(2)G*$	$(1+2n)M$	$(1+\ell)M$	$1M+2P$
Cui et al. [11]	$(1+2n_1)G*+G_T+nZ_p*$	$(1+2n_2)G*+(\ell)Z_p*$	$(2+2n_1)E+nM$	$(1+2n_2)E+\ell M$	$(1+2n_2)P+n_2E$
Xu et al. [28]	$((2+n')G*+G_T)n'$	$(2\ell)G*$	$(2n')E+n'P$	$(3\ell)E$	$4P+2(\ell-1)Pn''$
Wang et al. [27]	$(4+3n)G^*+(2)G_T^*$	$(7)G^*$	$(6)E$	$(7+3n)E$	$2P$
Zeng et al. [29]	$(n)G^*+(n)G_T^*+(n)C_{IBE}$	$(\ell)G^*+(\ell)S$	$(3n)E+(n)P+(n)C_{IBE}$	$(\ell)E+EC_{IBE}$	$(\ell n)P$
He et al. [39]	$(1+2n+2n_1)G^*$	$(1+2\ell+2n_2)G^*$	$(1+3n+2n_1)E$	$(3+2n_2+3\ell)E$	$(1+\sum_{i=1}^{\lvert A\rvert}A_i'+\sum_{j=1}^{\lvert W\rvert}W_j')E+(1+\sum_{i=1}^{\lvert A\rvert}Ai'+\sum_{j=1}^{\lvert W\rvert}Wj')P$
Han et al. [22]	$(1+n)G+G_T$	$(2\ell)G$	$(2+n)E$	$(3\ell)E$	$(\ell')E+(2\ell')P$
Lai et al. [23]	$(1+n)G+G_T$	$(2\ell)G$	$(2+2n)E$	$(4\ell)E$	$(\ell')E+(2\ell')P$
Lv et al. [24]	$(1+n)G+G_T$	$(2\ell)G$	$(2+2n)E$	$(4\ell)E$	$(\ell')E+(2\ell')P$
MWMR-BKSE	$(1+2n)G+G_T$	$(4\ell)G$	$(2+3n)E$	$(4\ell)E$	$(2\ell+\ell')E+(4\ell')P$

G, G_T: Size of element in composite order group G, G_T, G^*, G_T^*: Size of an element in prime order (p) group G^*, G_T^*, Z_p^*: Size of element in multiplicative group of prime order Z_p, n: Total keywords in a ciphertext, n': Average of total keywords in email, n'': Number of relative emails $(n'' << n)$, ℓ: Number of keywords in query, ℓ': $\max(|I|)$ where $|I| \in I'$ and I' is a set of minimum subsets satisfying the keyword query, C_{IBE}: Size of ciphertext computed by identity-based encryption(IBE), S: String of random size, n_1: Number of attributes in attribute policy associated with ciphertext, n_2: Number of attributes in attribute policy associated with token, $|A|$: Total number of subsets satisfying attribute policy, $|W|$: Total number of subsets satisfying keyword policy, P: Pairing, E: Exponentiation, M: Scalar multiplication, EC_{IBE}: Encryption overhead for IBE

6.2 Computational Overhead

We present computational overhead for *Encryption()*, *TokGen()*, and *Search()* algorithms in terms of the major operations viz. scalar multiplication (M), exponentiation (E), and pairing (P). From Table 4, it can be observed that the scheme [27] offers the most efficient *Encryption()* algorithm with constant computational overhead, i.e., ($6E$) as compared to the other listed schemes that demand n number of E or M operations for the same algorithm. However, with such an efficiency, [27] lacks multi-writer support in system. On the other hand, though the scheme [29] supports multiple writers and readers with $((3n)E + (n)P + (n)C_{IBE})$ operations, it suffers from the highest encryption overhead. Furthermore, compared to the other listed Boolean schemes [22–24], MWMR-BKSE has high encryption overhead. Such overhead is due to the provision of multi-writer multi-reader support in the system. In addition, though the recent Boolean scheme [28] supports multiple writers and readers, with $((2n')E + n'P)$ encryption overhead it serves to a single reader only (refer Sect. 6.3). On the other hand, with $((2+3n)E)$ operations, the proposed scheme has the encryption overhead similar to the recent scheme [39] having a single attribute in attribute policy associated with ciphertext (could be impractical)[3]. Thus, considering more than one attribute in attribute policy of ciphertext in [39], the proposed scheme is computationally more efficient than [39].

For *TokGen()* algorithm, among the listed Boolean keyword searchable schemes [22–24,27,28], the scheme [27] is considered as one of the most inefficient schemes since it requires n number of exponentiation operations for any query size (i.e., any value of ℓ). Such *TokGen()* algorithm is completely impractical for $n >>> \ell$. In contrast, the other schemes [11,22–24,28,29] including MWMR-BKSE offer *TokGen()* algorithm with computational overhead linear to ℓ. More precisely, with $((4\ell)E)$ operations, the computational overhead for the proposed *TokGen()* algorithm is almost same as that of the scheme [23]. However, this overhead is higher than [22,24,29] due to computation of additional component based on DR's secret search key.

In the case of *Search()* algorithm, the schemes [9,12] perform best with constant computational overhead. However, they support the conjunctive keyword query only. Among the Boolean keyword searchable schemes [22–24,27,28], the scheme [27] offers the most efficient *Search()* algorithm with constant computational overhead. On the other hand, the proposed *Search()* algorithm has a higher computational cost than the schemes [22–24,27,28]. Such overhead is due to the large-sized ciphertext and token generated by the proposed *Encryption()* and *TokGen()* algorithms, respectively. Further, the proposed *Search()* algorithm could be comparatively better than the scheme [39] having search overhead linear to the size of subsets satisfying attribute policy and keyword policy.

From the above discussion, we state that the proposed MWMR-BKSE offers Boolean keyword search along with multi-writer multi-reader setting at the moderate storage-computational overhead.

6.3 Overhead Considering Multi-reader Support

MWMR-BKSE supports multiple writers and multiple readers in the system. We show the comparative analysis of various Boolean searchable schemes [22–24,27–29] including MWMR-BKSE in Table 5. From table, we note that the ciphertext storage overhead for schemes [22–24,28,29] is $O(n \cdot R)$ since a separate ciphertext is computed for each reader using his public key. Consequently, encryption overhead rises to $O(n \cdot R)$ for R number of readers (users), in such schemes. In contrast, the proposed MWMR-BKSE can support multiple readers without any additional storage and computational overhead for ciphertexts in the system.

7 Performance Evaluation and Analysis

7.1 Experimental Setup

To empirically evaluate MWMR-BKSE, we simulate the proposed *Encryption()*, *TokGen()*, and *Search()* algorithms as well as the corresponding algorithms of the existing Boolean searchable schemes [22–24] having similar group structure (i.e., composite order groups). For implementation, we use Java Pairing-based Cryptographic library (JPBC) [41]. From the JPBC library, we utilize **Type A1** pairing ($G \times G \rightarrow G_T$) that is based on elliptic curve $E(F_q)$: $y^2 = x^3 + x$. Here, a group G is a composite order group of 4 subgroups, i.e., $G = G_{p1} \times G_{p2} \times G_{p3} \times G_{p4}$ where subgroup Gpi is a group of points of $E(F_{pi})$ for $1 \leq i \leq 4$. The order of G is $N = p1 \times p2 \times p3 \times p4$ and bit length of each pi is 256 bits. In addition, to conduct experiments, we use a client–server architecture where DR and DW are considered as clients whereas SS and ETA are considered as servers. In order to execute client-side algorithms (i.e., *Encryption()*, *TokGen()*), we use a local 32-bit, 2.10 GHz Pentium Core 2 Duo CPU with Windows 7 machine. For server-side algorithm (i.e., *Search()*) we employ an Amazon EC2 T2 (t2.micro) [42] instance of 64-bit, 3.3 GHz Intel Xeon processor with Microsoft Windows Server 2012. To realize the role of ETA, we use a local server class machine with 64-bit 3.10 GHz Intel Xeon processor and Microsoft Windows Server 2008 operating system.

For experimental analysis, we use the real-world dataset—Enron email dataset [43] that includes about 5M emails. We randomly choose emails from this dataset and use them as payload. To encrypt the payload, we employ standard AES algorithm (for 128 bit key) from javax.crypto package. We

Table 5 Overhead considering multi-reader support for Boolean searchable schemes

Schemes	Ciphertext size	Encryption overhead
Xu et al. 2019[28]	$(((2+n')G^* + G_T^*)n')R$	$((2n')E + n'P)R$
Wang et al. 2017 [27]	$(4+3n)G^* + 2G_T^*$	$6E$
Zeng et al. 2018 [29]	$(nG^* + nG_T^* + nC_{IBE})R$	$3nE + nP + EC_{IBE}$
He et al. 2018 [39]	$(1+2n+2n_1)G^*$	$(1+3n+2n_1)E$
Han et al. 2014 [22]	$((1+n)G + G_T)R$	$((2+n)R)E$
Lai et al. 2013 [23]	$((1+n)G + G_T)R$	$((2+2n)R)E$
Lv et al. 2014 [24]	$((1+n)G + G_T)R$	$((2+2n)R)E$
MWMR-BKSE	$(1+2n)G + G_T$	$(2+3n)E$

G, G_T: Size of an element in composite order group G, G_T, G^*, G_T^*: Size of an element in prime order (p') group G^*, G_T^*, n: Total keywords in ciphertext, n': Average of total keywords in email, n'': Number of relative emails ($n'' << n$), ℓ: Number of keywords in query, n_1: Number of attributes in attribute policy associated with ciphertext, C_{IBE}: Size of ciphertext computed by identity-based encryption (IBE), E: Exponentiation, R: Number of Readers, EC_{IBE}: Encryption overhead for IBE

Table 6 Simulation Parameters

Parameters	Values for simulation
n	{10, 25, 50, 75, 100}
ℓ	{10, 20, 30, 40, 50}
ℓ'	{10, 20, 30, 40, 50}
R	{100, 200, 300, 400, 500}

prepare keyword space $\mathcal{KS}$ of about $n = 100$ keyword fields relevant to the email system. We simulate the algorithms with respect to 4 significant parameters—(i) Number of keywords in ciphertexts (n), (ii) Number of keywords in Boolean query (ℓ), (iii) Number of keywords in the maximum sized set from the set of minimum subset I' (ℓ'), (iv) Number of Readers in system (R) (Table 6).

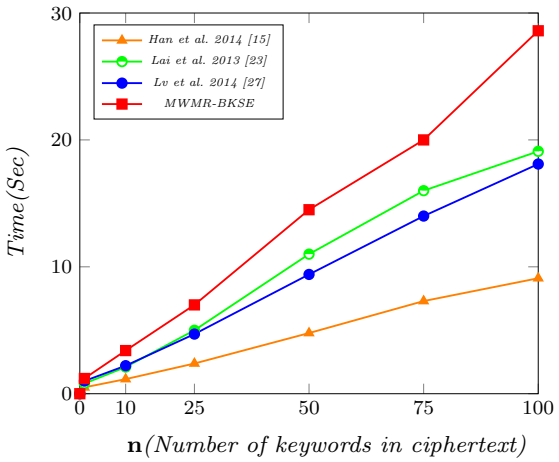


Fig. 4 Simulation results for Encryption() algorithm considering multi-keyword search

7.2 Result Analysis

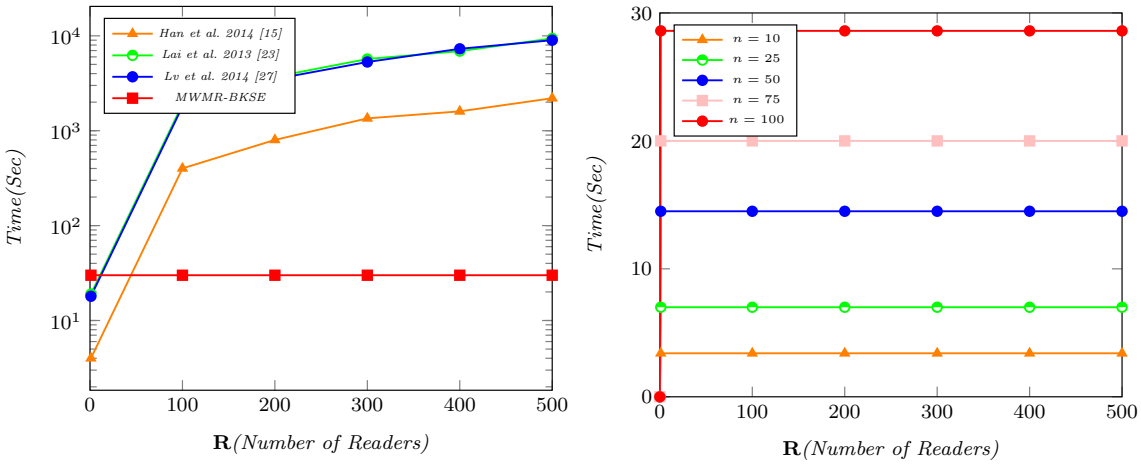
We execute the proposed *Encryption()*, *TokGen()*, and *Search()* algorithms multiple times and consider their average values as results. Based on the simulation parameters (Table 6), we evaluate the algorithms as follows:

Encryption() algorithm: We first analyze the performance of the proposed *Encryption()* algorithm by preparing ciphertexts with different number of keywords (i.e., n) (Fig. 4).

From results, we note that the execution time for *Encryption()* algorithm is linear to n as that in the schemes [22–24]. However, the proposed algorithm is the slowest among all schemes since it uses master public key as well as DW's secret key to encrypt each keyword involved in the ciphertext.

Furthermore, we show the encryption time overhead for different Boolean searchable encryption schemes including MWMR-BKSE under the existence of multiple readers in the system (Fig. 5a). We also show the effect of n on the proposed *Encryption()* algorithm in Fig. 5b. From Fig. 5a, it can be noticed that for the proposed algorithm, execution overhead remains constant irrespective of the number of readers (R) in system unlike the schemes [22–24] having encryption overhead linear to R. Such execution efficiency is achieved with the usage of a master public key in encryption of keywords instead of individual DR's public key as used by [22–24]. From Fig. 5b, we remark that in MWMR-BKSE, though the encryption overhead is constant, the number of keywords in ciphertext(n) affects the execution time. Such performance deviation is due to the fact that the encryption algorithm is required to be processed each keyword in ciphertext separately and hence increasing number of keywords in ciphertext would increase the overall execution time of *Encryption()* algorithm.

TokGen() algorithm: We execute the proposed *TokGen()* algorithm as well as the similar algorithm of the schemes [22–24] for different values of ℓ (Table 6) and present their



(a) For boolean searchable schemes

(b) MWMR-BKSE

Fig. 5 Simulation results for Encryption() considering multi-reader support

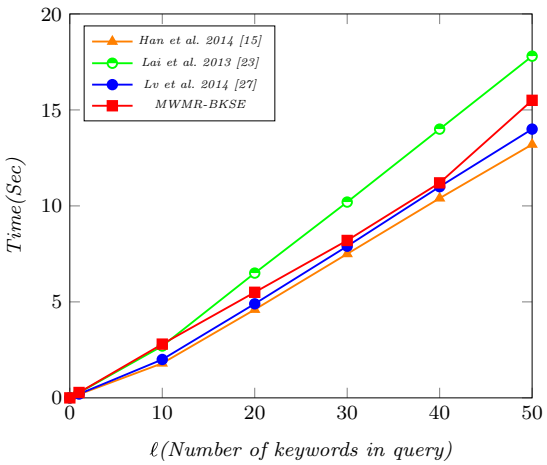


Fig. 6 Simulation results for TokGen() algorithm

execution time complexities in Fig. 6. As a large number of keywords makes a query complex and impractical, we select comparatively small values for ℓ.

From results, we identify that the proposed *TokGen()* algorithm executes in time linear to ℓ as that of [22–24]. Such overhead is due to the computational cost involved in encryption of each keyword in query using secret key (U_{RID}) of DR. Moreover, as that in [23], the proposed scheme takes execution time higher than the schemes of [22,24]. Such slow performance of proposed *TokGen()* algorithm is due to computation of the additional component based on DR's secret search key.

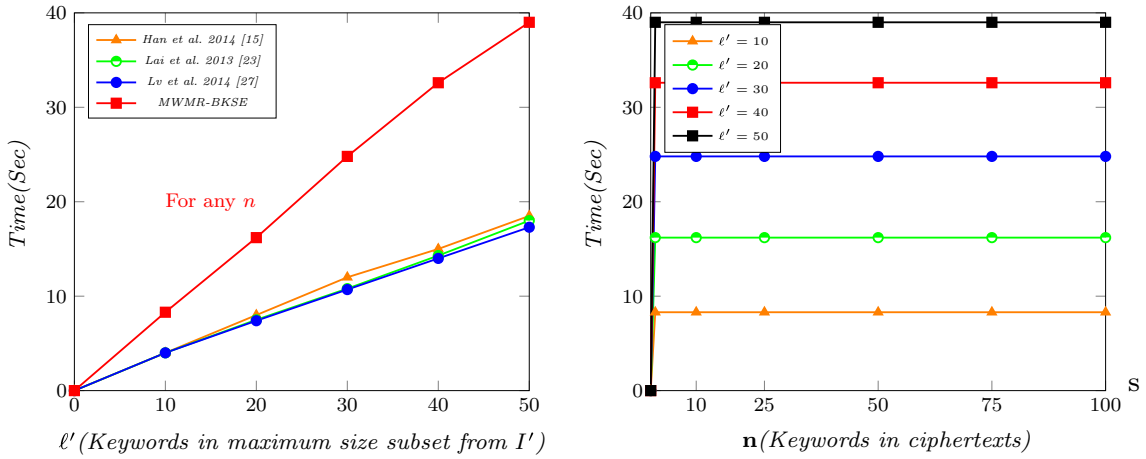
Search() algorithm: We consider the worst case scenario where a number of keywords in the maximum sized subset of the set of subsets (ℓ'), is same as the number of keywords in query (ℓ) that is, $\ell' = \ell$. We execute the proposed *Search()* algorithm as well as the similar algorithms of the schemes [22–24] for different values of ℓ' (Table 6) and present their results in Fig. 7a. From the result, we note that the time required to execute the proposed *Search()* algorithm is linear to ℓ' as that in [22–24]. However, the proposed scheme is slowest among all. Such performance degradation is due to the size of ciphertext and token (generated by the proposed algorithm) which is considerably larger than the size of ciphertext and token computed by the schemes [22–24] (as shown in Table 4).

We additionally show the search time overhead for the proposed scheme for different sized ciphertexts (i.e., with varied values of n) and different sized set of minimum subsets (i.e., with varied values of ℓ') (Fig. 7b). From results, we note that for different ℓ', the overhead involved in the proposed *Search()* algorithm is constant regardless of the number of keywords in ciphertexts (n). Such behavior is due to the fact that the proposed search algorithm applies ℓ' number of pairing and scalar multiplication operations onto ℓ' components of ciphertext. Thus, no matter whatever the size of ciphertext, the overall computational cost of the proposed *Search()* algorithm depends only on ℓ'.

From the above analysis, we state that our empirical results are following the theoretically measured computational complexity presented in Tables 4 and 5.

8 Concluding Remarks

The existing Boolean keyword searchable schemes either support multiple writers and multiple readers with large computational overhead along with weak security for ciphertexts or support a single writer and single reader with moderate computational overhead along with strong security for ciphertexts. With the proposed scheme, we offer a



(a) For boolean searchable schemes **(b)** MWMR-BKSE

Fig. 7 Simulation results for Search() algorithm

Boolean keyword searchable encryption to support multiple writers and multiple readers in system with moderate computational overhead as well as strong security for ciphertexts. The proposed multi-writer multi-reader Boolean Keyword Searchable Encryption (MWMR-BKSE), is a trusted authority-based scheme where any client can register with the system as a reader or writer at any time. The separate sets of readers and writers prepared in MWMR-BKSE avoid communication between writers and readers before data sharing. However, to offer such a multi-writer multi-reader setting, the trusted authority issues to each reader and writer a separate secret key whose size is comparatively larger than the key size issued to readers in the existing multi-reader schemes. In addition, a Boolean keyword search operation is offered by MWMR-BKSE utilizing key policy-based attribute-based encryption. Furthermore, the security of ciphertexts against chosen keyword attack is proved using full security model for the proposed scheme. From theoretical analysis and empirical evaluation, we conclude that the storage—computational overhead for MWMR-BKSE is higher than the existing Boolean keyword scheme for single-writer single-reader setting. Such overhead is due to the large sized keys issued to writers or readers that ultimately generates more components in ciphertexts or token and consumes more time in encryption or token generation. However, with such overhead, MWMR-BKSE offers the first Boolean keyword searchable encryption scheme supporting multiple writers and multiple readers in the system along with full security for ciphertext.

Appendix A

Lemma 1 *$Game_{real}$ and $Game_0$ are computationally indistinguishable, if $\mathbb{G}$ satisfies Assumption 1.*

Let an algorithm $\mathcal{A}$ distinguishes $Game_{real}$ and $Game_0$. Then, we can build an algorithm $\mathcal{B}$ with non-negligible advantage ϵ in breaking Assumption 1.

Proof Suppose $\mathcal{B}$ is given $\{g, X_3, X_4, T\}$ as per *Assumption 1* and it will simulate $Game_0$ or $Game_{real}$ with algorithm $\mathcal{A}$ as follows: **Setup** Let $\mathcal{B}$ randomly selects $\alpha, a_1, a_2, \ldots, a_n, a'_1, a'_2, \ldots, a'_n \in Z_N$, $g \in G_{p1}$, and $h_1, \ldots, h_n \in G_{p4}$. Then, for $1 \leq i \leq n$, $\mathcal{B}$ sets $u_i{=}g^{a_i}$, $H_i{=}u_i h_i$, and $u'_i = g^{a'_i}$. It then sets the public key pk and master secret key msk as follows:

$$pk = \{N, g, g^{\alpha}, H_i, X_4\}_{1 \leq i \leq n}$$
$$msk = \{\alpha, u_i, u'_i, X_3\}_{1 \leq i \leq n}.$$

From msk, $\mathcal{B}$ constructs the secret key for each data reader RID as follows:

- It selects a random $x_{RID} \in Z_N$ and computes $y_{RID} = \alpha / x_{RID}$.
- It selects a random $x'_{RID} \in Z_N$ and computes $u'_{i\_RID} = (u'_i)^{1/x'_{RID}}$ for $1 \leq i \leq n$.

Afterward, $\mathcal{B}$ sets a secret key for each user RID as

$$sk_{RID} = \{x_{RID}, u_i, u'_{i\_RID}\}_{1 \leq i \leq n}.$$

$\mathcal{B}$ also sets the corresponding server key for each READER RID as

$$sr_{RID} = \{\text{RID}, y_{RID}, x'_{RID}\}.$$

$\mathcal{B}$ maintains RList = $\{sr_{RID}\}$ for each registered user, at the server.

Phase 1. In response to $\mathcal{A}$'s token key request for a user RID, $\mathcal{B}$ generates a normal token key using *Token()* algorithm with parameter SK_{RID}.

Challenge $\mathcal{A}$ sends two sets of keywords, i.e., W_0 and W_1. $\mathcal{B}$ first chooses $b \in \{0, 1\}$ randomly and does the following.

- $\mathcal{B}$ selects random values $h, \tilde{Z}_0, \{\tilde{Z}_{1i}, \tilde{Z}'_{1i}\}_{1\le i\le n} \in G_{p4}$.
- Let $W_b = \{w_{b,1}, \ldots, w_{b,n}\}$. $\mathcal{B}$ computes

$$C_0 = e(g^{\alpha}, T), \quad C'_0 = T \cdot \tilde{Z}_0,$$
$$C_i = T^{a_i w_{\beta,i}} \cdot \tilde{Z}_{1i}, \quad C'_i = T^{a'_i} \cdot \tilde{Z}'_{1i}.$$

$\mathcal{B}$ sends the challenge ciphertext $C_{W_b} = \{C_0, C'_0, C_i, C'_i\}_{1\le i\le n}$ to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ can send other token key queries for the set of keywords $W \neq \{W_0, W_1\}$. $\mathcal{B}$ responds same as in *Phase 1*.

Guess. $\mathcal{A}$ outputs a guess b' of b.

- If $T \in G_{p1} \times G_{p2}$ Let $T = g^s \cdot g_2^c$, then the challenge ciphertext can be computed as follows:

$$C_0 = e(g, g^{\alpha})^s, \; C'_0 = (gh)^s \cdot Z_0 \cdot g_2^c,$$
$$C_i = (H_i^{w_{b,i}})^s Z_{1i} \cdot g_2^{c\gamma_i}, \; C'_i = (u'_i)^s \cdot Z'_{1i} \cdot {g_2}^{c\gamma'_i}$$

  where $Z_0 = \tilde{Z}_0 \cdot h^{-s}$, $Z_{1i} = \tilde{Z}_{1i} \cdot (h_i)^{-s w_{b,i}}$, $\gamma_i = a_i w_{b,i}$, $Z'_{1i} = \tilde{Z}'_{1i}$, $\gamma'_i = a'_i$.
  This is semi-functional ciphertext and $\mathcal{B}$ simulates $Game_0$. Since value of $a_i, a'_i, w_{b,i}$ modulo $p1$ are uncorrelated from their values modulo $p2$, it is properly distributed.
- If $T \in G_{p1}$. The challenge ciphertext is normal ciphertext and $\mathcal{B}$ simulates $Game_{real}$.

Thus, $\mathcal{B}$ can break *Assumption 1* with advantage ϵ, if $\mathcal{A}$ can distinguish two games, i.e., $Game_0$ and $Game_{real}$ with non-negligible advantage ϵ. □

Lemma 2 *$Game_{k-1,2}$ and $Game_{k,1}$ are computationally indistinguishable, if $\mathbb{G}$ satisfies Assumption 2.*

Let an algorithm $\mathcal{A}$ distinguishes $Game_{k-1,2}$ and $Game_{k,1}$. Then, we can build an algorithm $\mathcal{B}$ with non-negligible advantage ϵ in breaking Assumption 2.

Proof Suppose $\mathcal{B}$ is given $\{g, X_1X_2, Y_2Y_3, X_3, X_4, T\}$ as per *Assumption 2* and it will simulate $Game_{k-1,2}$ or $Game_{k,1}$ with algorithm $\mathcal{A}$ as follows:

Setup. Same as **LEMMA 1**.

Phase 1 The response of $\mathcal{B}$ for a token query request Q=$(A, \rho, T = \{t_{\rho(1)}, \ldots, t_{\rho(\ell)}\})$ for a user RID as follows:

- For $j < k$, $\mathcal{B}$ chooses a random vector v such that $1 \cdot v = x_{RID}$. It also selects a random vector z', random exponents $r_i \in Z_N$, random elements $V_{0i}, V_{1i}, V_{2i}, V_{3i} \in G_{p3}$. $\mathcal{B}$ then sets

$$D_{0i} = g^{A_i v} \cdot (Y_2Y_3)^{A_i z'} \cdot V_{0i}, \quad D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot V_{1i},$$
$$D_{2i} = (u'_{i\_RID})^{r_i} \cdot V_{2i}, \quad D_{3i} = g^{r_i} \cdot V_{3i}.$$

  This is properly distributed semi-functional token key of Type 2 because the value of $A_i z'$ modulo $p2$ is uncorrelated to its value modulo $p3$.
- For $j > k$, since $\mathcal{B}$ knows the secret key SK_{RID}, he can create a normal token key by running the *Token()* algorithm.
- For the kth token key, $\mathcal{B}$ chooses a random vector v' such that $1 \cdot v' = x_{RID}$, a random vector v' such that $v' \cdot 1 = 0$, a random exponents $\tilde{\eta_i} \in Z_N$, random elements $\tilde{V}_{0i}, \tilde{V}_{1i}, \tilde{V}_{2i}, \tilde{V}_{3i} \in G_{p3}$. It then sets:

$$D_{0i} = g^{A_i v} \cdot V_{0i}, \quad D_{1i} = T^{\tilde{\eta}_i a_{\rho(i)} t_{\rho(i)}} \cdot \tilde{V}_{1i},$$
$$D_{2i} = T^{\tilde{\eta}_i a'_{\rho(i)}/x'_{RID}} \cdot \tilde{V}_{2i}, \quad D_{3i} = T^{\tilde{\eta}_i} \cdot \tilde{V}_{3i}$$

  - If $T \in G_{p1} \times G_{p2} \times G_{p3}$, then let $T = g^r g_2^d V$ and thus

$$D_{0i} = g^{A_i v} \cdot V_{0i},$$
$$D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot g_2^{\eta_i \gamma_{\rho(i)}} \cdot V_{1i},$$
$$D_{2i} = (u'_i)^{r_i / x'_{RID}} \cdot g_2^{\eta_i \gamma_{\rho(i)'}} \cdot V_{2i},$$
$$D_{3i} = g^{r_i} \cdot g_2^{\eta_i} \cdot V_{3i},$$

    where $v = v' + zr, r_i = r\tilde{\eta}_i, \eta_i = d\tilde{\eta}_i, \gamma_{\rho(i)} = a_{\rho(i)} t_{\rho(i)}, \gamma'_{\rho(i)} = a'_{\rho(i)}/x'_{RID}$, $V_{0i} = V^{A_i z} \cdot \tilde{V}_{0i}$, $V_{1i} = V^{\tilde{\eta}_i a_i t_{\rho(i)}} \cdot \tilde{V}_{1i}$, $V_{2i} = V^{\gamma_{1i}} \cdot \tilde{V}_{2i}$, $V_{3,i} = V^{\tilde{\eta}_i} \cdot \tilde{V}_{3i}$.
    This is semi-functional key of Type 1. Since the value of $\tilde{\eta}_i, a_i, t_{\rho(i)}$ modulo $p1$ are uncorrelated from their values modulo $p2$, it is properly distributed.
  - If $T \in G_{p1} \times G_{p3}$, It is a properly distributed normal token key.

Challenge. $\mathcal{A}$ sends two sets of keywords, i.e., W_0 and W_1. $\mathcal{B}$ chooses $b \in \{0, 1\}$ randomly and does the following.

- $\mathcal{B}$ selects random values $h, \tilde{Z}_0, \{\tilde{Z}_{1i}, \tilde{Z}'_{1i}\}_{1\le i\le n} \in G_{p4}$.
- Let $W_\beta = \{w_{b,1}, \ldots, w_{b,n}\}$. $\mathcal{B}$ computes

$$C_0 = e(g^{\alpha}, X_1X_2), \quad C'_0 = (X_1X_2) \cdot \tilde{Z}_0,$$
$$C_i = (X_1X_2)^{a_i w_{b,i}} \cdot \tilde{Z}_{1i}, \quad C'_i = (X_1X_2)^{a'_i} \cdot \tilde{Z}'_{1i}$$

- $\mathcal{B}$ sets the challenge ciphertext $C_{W_b} = \{C_0, C'_0, C_i, C'_i\}_{1\le i\le n}\}$ and sends it to $\mathcal{A}$.

– If X_1X_2=$g^s g_2^c$, Then

$$C_0 = e(g, g^{\alpha})^s, \quad C_0' = (gh)^s \cdot Z_0 \cdot g_2^c,$$
$$C_i = (H_i^{w_{b,i}})^s \cdot Z_{1i} \cdot g_2^{c\gamma_i}, \quad C_i' = (u_i')^s \cdot g_2{}^{c\gamma_i'} \cdot Z_{1i}'$$

where $Z_0 = \tilde{Z}_0 \cdot h^{-s}$, $Z_{1i} = \tilde{Z}_{1i}(h_i)^{-sw_{b,i}}$, $\gamma_i = a_i w_{b,i}$, $\gamma_i' = a_i'$, $Z_{1i}' = \tilde{Z}_{1i}'$.

This is semi-functional ciphertext. Since value of a_i, a_i', $w_{b,i}$ modulo $p1$ are uncorrelated from their values modulo $p2$, it is properly distributed.

Phase 2. $\mathcal{A}$ can send other token key queries for the set of keywords $W \neq \{W_0, W_1\}$. $\mathcal{B}$ responds same as in *Phase 1*.

– If $T \in G_{p1} \times G_{p2} \times G_{p3}$, the kth key is properly distributed semi-functional key of Type 1. Thus, $\mathcal{B}$ has properly simulated $Game_{k,1}$.
– If $T \in G_{p1} \times G_{p3}$, the kth key is properly distributed normal key. Thus, $\mathcal{B}$ has properly simulated $Game_{k-1,2}$.

Thus, if $\mathcal{A}$ can distinguish two games, i.e., $Game_{k,1}$ and $Game_{k-1,2}$ with non-negligible advantage ϵ, $\mathcal{B}$ can break *Assumption 2* with advantage ϵ. □

Lemma 3 *$Game_{k,1}$ and $Game_{k,2}$ are computationally indistinguishable, if $\mathbb{G}$ satisfies Assumption 2.*

Let an algorithm $\mathcal{A}$ distinguishes $Game_{k,1}$ and $Game_{k,2}$. Then, we can build an algorithm $\mathcal{B}$ with non-negligible advantage ϵ in breaking Assumption 2.

Proof Suppose $\mathcal{B}$ is given $\{g, X_1X_2, Y_2Y_3, X_3, X_4, T\}$ as per Assumption 2 and it will simulate $Game_{k,1}$ or $Game_{k,2}$ with algorithm $\mathcal{A}$ as follows:

Setup. Same as in *LEMMA 1*.

Phase 1 The response of $\mathcal{B}$ for a token query request Q=$(A, \rho, T = \{t_{\rho(1)}, \ldots, t_{\rho(\ell)}\})$ for a user RID as follows:

– For $j < k$ and $j > k$, the response is same as in LEMMA 2.
– For the kth query, $\mathcal{B}$ chooses a random vector v such that $1 \cdot v = x_{RID}$, a random vector z, random exponents $\tilde{\eta}_i \in Z_N$, random elements $\tilde{V}_{0i}, \tilde{V}_{1i}, \tilde{V}_{2i}, \tilde{V}_{3i} \in G_{p3}$. $\mathcal{B}$ then sets:

$$D_{0i} = g^{A_i \cdot v}(Y_2Y_3)^{A_i z} \cdot \tilde{V}_{0i}, \quad D_{1i} = T^{\tilde{\eta}_i a_{\rho(i)} t_{\rho(i)}} \cdot \tilde{V}_{1i},$$
$$D_{2i} = T^{\tilde{\eta}_i a'_{\rho(i)}/x'_{RID}} \cdot \tilde{V}_{2i}, \quad D_{3i} = T^{\tilde{\eta}_i} \cdot \tilde{V}_{3i}$$

  – If $T \in G_{p1} \times G_{p2} \times G_{p3}$, then let $T = g^r g_2^d V$ and thus

$$D_{0i} = g^{A_i v} \cdot Y_2{}^{A_i z} \cdot V_{0i},$$
$$D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot V_{1i} \cdot g_2^{\eta_i \gamma_{\rho(i)}},$$
$$D_{2i} = (u_i')^{r_i/x_{RID}} \cdot V_{2i} \cdot g_2{}^{\eta_i \gamma'_{\rho(i)}},$$
$$D_{3i} = g^{r_i} \cdot V_{3i} \cdot g_2{}^{\eta_i}$$

where $r_i = r\tilde{\eta}_i$, $\eta_i = d\tilde{\eta}_i$, $\gamma_{\rho(i)} = a_{\rho(i)}t_{\rho(i)}$, $\gamma'_{\rho(i)} = a'_{\rho(i)}/x'_{RID}$, $V_{0i} = Y_3{}^{A_i z} \cdot \tilde{V}_{0i}$, $V_{1i} = V^{\tilde{\eta}_i a_i t_{\rho(i)}} \cdot \tilde{V}_{1i}$, $V_{2i} = V^{\tilde{\eta}_i a_i'/x'_{RID}} \cdot \tilde{V}_{2i}$, $V_{3i} = V^{\tilde{\eta}_i} \cdot \tilde{V}_{3i}$.
This is semi-functional key of Type 1. Since the value of $\tilde{\eta}_i, a_i, a_i', t_{\rho(i)}$ modulo $p1$ are uncorrelated from their values modulo $p2$, it is properly distributed.
  – If $T \in G_{p1} \times G_{p3}$, then it is properly distributed semi-functional key of Type 2.

Challenge. Same as *LEMMA 2*.

Phase 2. $\mathcal{A}$ can send other token key queries for the set of keywords $W \neq \{W_0, W_1\}$. $\mathcal{B}$ responds same as in *Phase 1*.

– If $T \in G_{p1} \times G_{p2} \times G_{p3}$, the kth key is properly distributed semi-functional key of Type 1. Thus, $\mathcal{B}$ has properly simulated $Game_{k,1}$.
– If $T \in G_{p1} \times G_{p3}$, the kth key is properly distributed semi-functional key of Type 2. Thus, $\mathcal{B}$ has properly simulated $Game_{k,2}$.

Thus, if $\mathcal{A}$ can distinguish two games, i.e., $Game_{k,1}$ and $Game_{k,2}$ with non-negligible advantage ϵ, $\mathcal{B}$ can break *Assumption 2* with advantage ϵ. □

Lemma 4 *$Game_{Final_{\zeta-1}}$ and $Game_{Final_{\zeta}}$ are computationally indistinguishable, if $\mathbb{G}$ satisfies Assumption 3.*

Let an algorithm $\mathcal{A}$ distinguishes $Game_{Final_{\zeta-1}}$ and $Game_{Final_{\zeta}}$. Then, we can build an algorithm $\mathcal{B}$ with non-negligible advantage ϵ in breaking Assumption 3.

Proof Suppose $\mathcal{B}$ is given $\{g, g_2, X_2, uh', u'h', g^s B_{24}, X_3, X_4, T\}$ as per Assumption 3 and it will simulate $Game_{Final_{\zeta-1}}$ and $Game_{Final_{\zeta}}$ with algorithm $\mathcal{A}$ as follows:

Setup. $\mathcal{B}$ randomly selects α, a_1, a_2, …, $a_{\max\{1,\zeta-1\}}$, $a_{\zeta+1}$,…, a_n, a_1', a_2', …, $a'_{\max\{1,\zeta-1\}}$, $a'_{\zeta+1}$, …, $a'_n \in Z_N$, and h_1,…, $h_{\max\{1,\zeta-1\}}$, $h_{\zeta+1}$, …, $h_n \in G_{p4}$. Then, it sets $u_1 = g^{a_1}$, …, $u_{\max\{1,\zeta-1\}} = g^{a_{\max\{1,\zeta-1\}}}$, $u_{\zeta+1} = g^{a_{\zeta+1}}$, …, $u_n = g^{a_n}$. It also sets $u_1' = g^{a_1'}$,…, $u'_{\max\{1,\zeta-1\}} = g^{a'_{\max\{1,\zeta-1\}}}$, $u'_{\zeta+1} = g^{a'_{\zeta+1}}$,…, $u_n' = g^{a'_n}$. $\mathcal{B}$ then computes $H_1 = u_1h_1$,…,$H_{\max\{1,\zeta-1\}} = u_{\max\{1,\zeta-1\}}h_{\max\{1,\zeta-1\}}$, $H_\zeta = uh'$, $H_{\zeta+1} = u_{\zeta+1}h_{\zeta+1}$,…, $H_n = u_nh_n$. It then sets a public key pk and a master secret key msk as follows:

$$pk = \{N, g, g^{\alpha}, H_i, X_4\}_{1 \le i \le n}, msk = \{\alpha, u_i, u_i', X_3\}_{1 \le i \le n}.$$

Additionally, $\mathcal{B}$ sets a secret key (sk_{RID}) for each user RID and the corresponding server key (sr_{RID}) same as in *LEMMA 1*. □

Phase 1 The response of $\mathcal{B}$ for $\mathcal{A}$'s token query request Q=$(A, \rho, T = \{t_{\rho(1)}, \ldots, t_{\rho(\ell)}\})$ for a reader RID as follows:

- $\mathcal{B}$ chooses a random vector v such that $1 \cdot v = x_{RID}$.
- It then selects a random vector z, random exponents $r_i \in Z_N$, random elements $\tilde{V}_{0i}, \tilde{V}_{1i}, \tilde{V}_{2i}, \tilde{V}_{3i} \in G_{p3}$ and sets: $D_{0i} = g^{A_i v} \cdot \tilde{V}_{0i} \cdot g_2^{A_i z}, \quad D_{3i} = g^{r_i} \cdot \tilde{V}_{3i}$
- If $\rho(i) = \zeta$ Then $\mathcal{B}$ sets $D_{1i} = (uX_2)^{r_i t_{\rho(i)}} \cdot \tilde{V}_{1i}$, $D_{2i} = (u'X_2)^{r_i / x'_{RID}} \cdot \tilde{V}_{2i}$
- Otherwise (i.e., $\rho(i) \neq \zeta$), $\mathcal{B}$ sets $D_{1i} = (g^{a_{\rho(i)}})^{r_i t_{\rho(i)}} \cdot \tilde{V}_{1i}$, $D_{2i} = (g^{a'_{\rho(i)}})^{r_i} \cdot \tilde{V}_{2i}$.

Thus, for $\rho(i) = \zeta$, the token components can be written as

$$D_{0i} = g^{A_i v} \cdot g_2^{\delta_{0i}} \cdot V_{0i}, \quad D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot g_2^{\delta_{1i}} \cdot V_{1i},$$
$$D_{2i} = (u')^{r_i / x'_{RID}} \cdot g_2^{\delta_{2i}} \cdot V_{2i}, \quad D_{3i} = g^{r_i} \cdot V_{3i}$$

where $\delta_{0i} = A_i z$, $\delta_{1i} = r_i t_{\rho(i)} \log_{g_2} X_2$, $\delta_{2i} = r_i / x'_{RID} \cdot \log_{g_2} X_2$, $V_{0i} = \tilde{V}_{0i}$, $V_{1i} = \tilde{V}_{1i}$, $V_{2i} = \tilde{V}_{2i}$,$V_{3i} = \tilde{V}_{3i}$

And for $\rho(i) \neq \zeta$

$$D_{0i} = g^{A_i v} \cdot g_2^{\delta_{0i}} \cdot V_{0i}, \quad D_{1i} = (u_{\rho(i)})^{r_i t_{\rho(i)}} \cdot V_{1i},$$
$$D_{2i} = (u')^{r_i / x'_{RID}} \cdot V_{2i}, \quad , D_{3i} = g^{r_i} \cdot V_{3i}$$

where $\delta_{0i} = A_i z$, $V_{0i} = \tilde{V}_{0i}$, $V_{1i} = \tilde{V}_{1i}$, $V_{2i} = \tilde{V}_{2i}$,$V_{3i} = \tilde{V}_{3i}$

This is properly distributed token key of Type 2.

Challenge. $\mathcal{A}$ sends two sets of keywords, i.e., W_0 and W_1. $\mathcal{B}$ first chooses $b \in \{0, 1\}$ randomly and does the following.

- $\mathcal{B}$ selects random values $h, \tilde{Z}_0, \{\tilde{Z}_{1i}, \tilde{Z}'_{1i}\}_{1 \leq i \leq n} \in G_{p4}$.
- It also selects random elements $C_1, \ldots, C_{\max\{1,\zeta-1\}}, C'_1, \ldots, C'_{\max\{1,\zeta-1\}} \in G_{p1} \times G_{p2} \times G_{p4}$.
- Let $W_b = \{w_{b,1}, \ldots, w_{b,n}\}$.
- $\mathcal{B}$ computes $C_0 = e(g, g^s B_{24}), \quad C'_0 = g^s B_{24} \cdot \tilde{Z}_0$,
- For $i = \zeta$, $\mathcal{B}$ computes $C_\zeta = (g^s B_{24})^{w_{b,\zeta}} \cdot T \cdot \tilde{Z}_{1\zeta}$, $C'_\zeta = (g^s B_{24}) \cdot T \cdot \tilde{Z}'_{1\zeta}$,
- For $\zeta \leq i \leq n$, it computes $C_i = (g^s B_{24})^{a_i w_{b,i}} \cdot \tilde{Z}_{1i}$, $C'_i = (g^s B_{24})^{a'_i} \cdot \tilde{Z}'_{1i}$
- $\mathcal{B}$ sets the challenge ciphertext $C_{W_b} = \{C_0, C'_0, \{C_i, C'_i\}_{1 \leq i \leq n}\}$ and sends it to $\mathcal{A}$.

Phase 2 $\mathcal{A}$ can send other token key queries for the set of keywords $W \neq \{W_0, W_1\}$. $\mathcal{B}$ responds same as in *Phase 1*.

Guess $\mathcal{A}$ outputs a guess b' of b.

- Let B_2, B_4 be the G_{p2}, G_{p4} parts of B_{24}, respectively, and D_2, D_4 be the G_{p2}, G_{p4} parts of D_{24}.
- Let $T = (u^s D_{24}, u' D_{24})$, then the challenge ciphertext can be computed as follows:

  $C_0 = e(g, g^\alpha)^s, \quad C'_0 = (gh)^s \cdot Z_0 \cdot g_2^c$

  where $Z_0 = h^{-s} \cdot B_4 \cdot \tilde{Z}_0$, $c = \log_{g_2} B_2$.
- For $i = \zeta$, $C_i = (H_i^{w_{b,i}})^s \cdot Z_{1i} \cdot g_2^{c\gamma_i}, \quad C'_i = u_i'^s \cdot Z'_{1i} \cdot g_2^{c\gamma'_i}$ where $c = \log_{g_2} B_2$, $Z_{1i} = B_4^{w_{b,\zeta}} \cdot D_4 \cdot \tilde{Z}_{1\zeta} (uh')^{-s w_{b,\zeta}}$, $\gamma_i = (\log_{g_2}(B_2^{w_{b,\zeta}} D_2))/c$, $\gamma'_i = (\log_{g_2}(B_2 D_2))/c$.
- For $\zeta < i \leq n$, $C_i = (H_i^{w_{b,i}})^s \cdot Z_{1i} \cdot g_2^{c\gamma_i}$, $C'_i = u'^s \cdot Z'_{1i} \cdot g_2^{c\gamma'_i}$ where $c = \log_{g_2} B_2$, $Z_{1i} = B_4^{a_i w_{b,i}} \cdot \tilde{Z}_{1i} \cdot (h)^{-s w_{b,i}}$, $\gamma_i = a_i w_{b,i}$, $Z'_{1i} = B_4^{a'_i} \cdot \tilde{Z}'_{1i}$.
  Since the value of $a_i, a'_i, w_{b,i}$ modulo $p1$ are uncorrelated from their values modulo $p2$, it is properly distributed semi-functional ciphertext with $C_1, \ldots, C_{\max\{1,\zeta-1\}}, C'_1, \ldots, C'_{\max\{1,\zeta-1\}}$ random in $G_{p1} \times G_{p2} \times G_{p4}$. Thus, $\mathcal{B}$ has properly simulated $Game_{Final_{\zeta-1}}$.
- If $T \in G_{p1} \times G_{p2} \times G_{p4}$, then this is properly distributed semi-functional ciphertext with $C_1, \ldots, C_{\max\{1,\zeta\}} C'_1, \ldots, C'_{\max\{1,\zeta-1\}}$ random in $G_{p1} \times G_{p2} \times G_{p4}$. Thus, $\mathcal{B}$ has properly simulated $Game_{Final_\zeta}$.

Thus, if $\mathcal{A}$ can distinguish two games, i.e., $Game_{Final_{\zeta-1}}$ and $Game_{Final_\zeta}$ with non-negligible advantage ϵ, $\mathcal{B}$ can break *Assumption 3* with advantage ϵ.

Appendix B

Lewko et al. [44] have defined a method to convert a Boolean formula into linear secret sharing scheme (LSSS) matrix. The authors in [44] consider a Boolean formula as an access tree structure where all the internal nodes are either 'AND' or 'OR' gates and the leaf nodes are attributes. Additionally, they take a vector (1, 0, 0, …, 0) as the sharing vector for LSSS matrix. From [44], we define a procedure *Convert()* to prepare LSSS matrix from the access tree of a Boolean query of keywords.

Procedure Convert (T,A)

Input: An access tree T of depth d (0 to $d-1$ levels). *Output*: An LSSS matrix A of ℓ rows where ℓ is the number of keywords in Boolean query assuming (i) no keyword repetition in query, (ii) each keyword Key_k has a fixed row number in A ($1 \leq k \leq \ell$).

Let N_{ij} is the ith node at jth level of T, $Left(N_{ij})$ is a left child of node N_{ij}, $Right(N_{ij})$ is a right child of node N_{ij}, V_{ij} is a vector assigned to node N_{ij}, $VLen(V_{ij})$ is a length of vector V_{ij}, $Type_{ij}$ is the type (i.e., AND, OR, Leaf) of node N_{ij}, TN_j is total number of nodes at jth level. Let Key_{ij} is the keyword associated with ith 'Leaf' node at jth level. Let c is a global counter variable.

1. $c \leftarrow 1$, $flag \leftarrow 0$, $V_{ij} \leftarrow (1)$ (a vector of length 1).
2. For $j = 0$ to $j = d - 1$ do
3. For $i = 1$ to $i = TN_j$ do
4. If $Type_{ij}$ is 'OR' then
   - Set $Left(N_{ij}) \leftarrow V_{ij}$
   - Set $Right(N_{ij}) \leftarrow V_{ij}$
5. If $Type_{ij}$ is 'AND' then
   - Set $v1 \leftarrow V_{ij}$
   - If $VLen(v1) \neq c$ Then
   - Append 0 to $v1$ till $VLen(v1) \leftarrow c$
   - Set $Left(N_{ij}) \leftarrow (v1|1)$ where | denotes concatenation.
   - Prepare $v2 \leftarrow (0, 0, \ldots, 0)$ such that $VLen(v2) = c$
   - Set $Right(N_{ij}) \leftarrow (v2| - 1)$, $flag \leftarrow 1$
6. If $Type_{ij}$ is 'Leaf' and $Key_{ij} = K_k$ Then
   - Set $A[k] \leftarrow V_{ij}$
7. Set $i \leftarrow i + 1$
8. EndFor
9. If $(flag = 1)$ then
   - Set $c \leftarrow c + 1$, $flag \leftarrow 0$
10. Set $j \leftarrow j + 1$
11. EndFor.

References

1. Bao, F.; Deng, R.H.; Ding, X.; Yang, Y.: Private query on encrypted data in multi-user settings. In: International Conference on Information Security Practice and Experience, pp. 71–85. Springer, Berlin (2008)
2. Huang, H.; Du, J.; Wang, H.; Wang, R.: A multi-keyword multi-user searchable encryption scheme based on cloud storage. In: Trustcom/BigDataSE/I SPA, 2016 IEEE, pp. 1937–1943. IEEE (2016)
3. Hwang, Y.H.; Lee, P.J.: Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: Takagi, T., Okamoto, E., Okamoto, T., Okamoto, T. (eds.) Pairing-Based Cryptography—Pairing 2007, pp. 2–22. Springer, Berlin (2007)
4. Kiayias, A.; Oksuz, O.; Russell, A.; Tang, Q.; Wang, B.: Efficient encrypted keyword search for multi-user data sharing. In: European Symposium on Research in Computer Security, pp. 173–195. Springer, Berlin (2016)
5. Li, J.; Chen, X.: Efficient multi-user keyword search over encrypted data in cloud computing. Comput. Inform. **32**(4), 723–738 (2013)
6. Wang, S.; Zhang, X.; Zhang, Y.: Efficiently multi-user searchable encryption scheme with attribute revocation and grant for cloud storage. PLoS ONE **11**(11), e0167157 (2016)
7. Ye, J.; Wang, J.; Zhao, J.; Shen, J.; Li, K.C.: Fine-grained searchable encryption in multi-user setting. Soft Comput. **21**, 1–12 (2016)
8. Zhang, Y.; Liu, L.; Wang, S.: Multi-user and keyword-based searchable encryption scheme. In: 2016 12th International Conference on Computational Intelligence and Security (CIS), pp. 223–227. IEEE (2016)
9. Sharma, D.; Jinwala, D.C.: Multi-User searchable encryption with token freshness verification (MUSE-TFV). Secur. Commun. Netw. **2017**, 16 (2017)
10. Xu, L.; Xu, C.; Liu, J.K.; Zuo, C.; Zhang, P.: Building a dynamic searchable encrypted medical database for multi-client. Inf. Sci. **527**, 394–405 (2019)
11. Cui, Y.; Gao, F.; Shi, Y.; Yin, W.; Panaousis, E.; Liang, K.: An efficient attribute-based multi-keyword search scheme in encrypted keyword generation. IEEE Access **8**, 99024–99036 (2020)
12. Sharma, D.; Jinwala, D.C.: Multi-writer multi-reader conjunctive keyword searchable encryption. Int. J. Inf. Comput. Secur. (2020) bf (in press)
13. Ballard, L.; Kamara, S.; Monrose, F.: Achieving efficient conjunctive keyword searches over encrypted data. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) Information and Communications Security, pp. 414–426. Springer, Berlin (2005)
14. Boneh, D.; Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) Theory of Cryptography, pp. 535–554. Springer, Berlin (2007)
15. Byun, J.W.; Lee, D.H.; Lim, J.: Efficient conjunctive keyword search on encrypted data storage system. In: Atzeni, A.S., Lioy, A. (eds.) Public Key Infrastructure, pp. 184–196. Springer, Berlin (2006)
16. Chen, Z.; Wu, C.; Wang, D.; Li, S.: Conjunctive keywords searchable encryption with efficient pairing, constant ciphertext and short trapdoor. In: Chau, M., Wang, G.A., Yue, W.T., Chen, H. (eds.) Pacific–Asia Workshop on Intelligence and Security Informatics, pp. 176–189. Springer, Berlin (2012)
17. Ding, M.; Gao, F.; Jin, Z.; Zhang, H.: An efficient public key encryption with conjunctive keyword search scheme based on pairings. In: 2012 3rd IEEE International Conference on Network Infrastructure and Digital Content, pp. 526–530. IEEE (2012)
18. Hwang, M.S.; Hsu, S.T.; Lee, C.C.: A new public key encryption with conjunctive field keyword search scheme. Inf. Technol. Control **43**(3), 277–288 (2014)
19. Park, D.J.; Kim, K.; Lee, P.J.: Public key encryption with conjunctive field keyword search. In: Lim, C.H., Yung, M. (eds.) Information Security Applications, pp. 73–86. Springer, Berlin (2005)
20. Wang, P.; Wang, H.: Pieprzyk: keyword field-free conjunctive keyword searches on encrypted data and extension for dynamic groups. In: International Conference on Cryptology and Network Security, pp. 178–195. Springer (2008)
21. Zhang, B.; Zhang, F.: An efficient public key encryption with conjunctive-subset keywords search. J. Netw. Comput. Appl. **34**(1), 262–267 (2011)
22. Han, F.; Qin, J.; Zhao, H.; Hu, J.: A general transformation from KP-ABE to searchable encryption. Future Gener. Comput. Syst. **30**, 107–115 (2014)
23. Lai, J.; Zhou, X.; Deng, R.H.; Li, Y.; Chen, K.: Expressive search on encrypted data. In: Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, pp. 243–252. ACM (2013)
24. Lv, Z.; Hong, C.; Zhang, M.; Feng, D.: Expressive and secure searchable encryption in the public key setting (full version). In: IACR Cryptology ePrint Archive 2014, p. 614 (2014)
25. Kermanshahi, S.K.; Liu, J.K.; Steinfeld, R.: Multi-user cloud-based secure keyword search. In: Australasian Conference on Information Security and Privacy, pp. 227–247. Springer, Berlin (2017)
26. Sun, S.F.; Liu, J.K.; Sakzad, A.; Steinfeld, R.; Yuen, T.H.: An efficient non-interactive multi-client searchable encryption with support for Boolean queries. In: European Symposium on Research in Computer Security, pp. 154–172. Springer (2016)
27. Wang, Y.; Wang, J.; Sun, S.F.; Liu, J.K.; Susilo, W.; Chen, X.: Towards multi-user searchable encryption supporting Boolean

query and fast decryption. In: International Conference on Provable Security, pp. 24–38. Springer (2017)
28. Xu, P.; Tang, S.; Xu, P.; Wu, Q.; Hu, H.; Susilo, W.: Practical multi-keyword and Boolean search over encrypted e-mail in cloud server. IEEE Trans. Serv. Comput. (2019). https://doi.org/10.1109/TSC.2019.2903502
29. Zeng, M.; Zhang, K.; Qian, H.; Chen, X.; Chen, J.: A searchable asymmetric encryption scheme with support for Boolean queries for cloud applications. Comput. J. **62**(4), 563–578 (2018)
30. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98. ACM (2006)
31. Lewko, A.; Okamoto, T.; Sahai, A.; Takashima, K.; Waters, B.: Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Advances in Cryptology—EUROCRYPT 2010, pp. 62–91. Springer (2010)
32. Song, D.X.; Wagner, D.; Perrig, A.: Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy, 2000. S&P 2000. Proceedings, pp. 44–55. IEEE (2000)
33. Goh, E.J.; et al.: IACR Cryptology ePrint Archive. In: Secure indexes 2003, p. 216 (2003)
34. Chang, Y.C.; Mitzenmacher, M.: Privacy preserving keyword searches on remote encrypted data. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) Applied Cryptography and Network Security, pp. 442–455. Springer, Berlin (2005)
35. Boneh, D.; Di Crescenzo, G.; Ostrovsky, R.; Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) Advances in Cryptology-Eurocrypt 2004, pp. 506–522. Springer, Berlin (2004)
36. Katz, J.; Sahai, A.; Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Advances in Cryptology—EUROCRYPT 2008, pp. 146–162. Springer, Berlin (2008)
37. Wang, P.; Wang, H.: Pieprzyk: threshold privacy preserving keyword searches. In: International Conference on Current Trends in Theory and Practice of Computer Science, pp. 646–658. Springer (2008)
38. Wang, P.; Wang, H.; Pieprzyk, J.: Common secure index for conjunctive keyword-based retrieval over encrypted data. In: Secure Data Management, pp. 108–123 (2007)
39. He, K.; Guo, J.; Weng, J.; Weng, J.; Liu, J.K.; Yi, X.: Attribute-based hybrid Boolean keyword search over outsourced encrypted data. IEEE Trans. Dependable Secure Comput. (2018). https://doi.org/10.1109/TDSC.2018.2864186
40. Beimel, A.: Secure schemes for secret sharing and key distribution. Ph.D. thesis, Technion-Israel Institute of technology, Faculty of Computer Science (1996)
41. De Caro, A.; Iovino, V.: jpbc: Java pairing based cryptography. In: Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011, pp. 850–855. IEEE, Kerkyra, Corfu, Greece, June 28–July 1 (2011)
42. Miller, F.P.; Vandome, A.F.; McBrewster, J.: Amazon Web Services. Alpha Press (2010)
43. Cohen, W.W.: Enron email dataset (2009). https://www.cs.cmu.edu/~enron/
44. Lewko, A.; Waters, B.: Decentralizing attribute-based encryption. In: Advances in Cryptology—EUROCRYPT 2011, pp. 568–588. Springer (2011)