Novel Multi-user Conjunctive Keyword Search Against Keyword Guessing Attacks Under Simple Assumptions

Zhiyuan Zhao[1], Jianhua Wang[1]
[1] Zhengzhou Information Science and Technology Institute
Zhengzhou 450001, Henan – P. R. China
[e-mail: zzy_taurus@foxmail.com]
*Corresponding author: Zhiyuan Zhao

Received December 11, 2016; revised March 2, 2017; accepted March 11, 2017; published July 31, 2017

Abstract

Conjunctive keyword search encryption is an important technique for protecting sensitive personal health records that are outsourced to cloud servers. It has been extensively employed for cloud storage, which is a convenient storage option that saves bandwidth and economizes computing resources. However, the process of searching outsourced data may facilitate the leakage of sensitive personal information. Thus, an efficient data search approach with high security is critical. The multi-user search function is critical for personal health records (PHRs). To solve these problems, this paper proposes a novel multi-user conjunctive keyword search scheme (mNCKS) without a secure channel against keyword guessing attacks for personal health records, which is referred to as a secure channel-free mNCKS (SCF-mNCKS). The security of this scheme is demonstrated using the Decisional Bilinear Diffie-Hellman (DBDH) and Decision Linear (D-Linear) assumptions in the standard model. Comparisons are performed to demonstrate the security advantages of the SCF-mNCKS scheme and show that it has more functions than other schemes in the case of analogous efficiency.

Keywords: Personal health records, Conjunctive keyword search, Search policy, Channel free, Keyword guessing attacks, Multi-user

This work was supported by the National Key Research Program of China, "Collaborative Precision Positioning Project"(No. 2016YFB0501900).

1. Introduction

Cloud computing has the advantages of dynamic expansion, on-demand service and billing quantity; this computing resource represents one of the most important information technology revolutions since the invention of the Internet [1]. With the development of cloud computing, personal health records (PHRs) have emerged as a patient-centric model of health information exchange, and architectures for storing PHRs in the cloud have been proposed [2]. However, numerous issues regarding PHR services must be investigated: 1. How can communication be secured between different PHR clouds? To resolve this problem, Amjad Mehmood et al. [3] proposed a secure multi-agent-based framework for communication among open clouds. In their framework, each cloud has a secure mobile agent that is responsible for secure communication among the cloud servers. 2. How can authentication technology be integrated into the cloud environment to protect PHRs? To resolve this problem, Ismail Butun et al. [4] proposed cloud-centric, multi-level authentication as a service approach that addresses scalability and time constraints and demonstrated its effectiveness. 3. How can the problem of information wireless transmission among a large number of users be resolved? To resolve this problem, Mohammad Shojafar et al. [5] applied learning automata for channel assignment in multi-radio wireless mesh networks (WMNs).

A PHR service enables a patient to create, manage, and control his or her personal health data in one location accessed via the web; this approach has facilitated the efficient storage, retrieval, and sharing of medical information. However, due to the high cost of building and maintaining specialized data centres, many PHR services are outsourced or provided by third-party service providers. To guarantee the security of their sensitive personal health information (PHI), patients should adopt encryption for their PHI before storing it in the cloud.

However, encryption may severely hinder several functionalities that users are accustomed to receiving from cloud solutions. For instance, searches for a data owner's outsourced encrypted data would not be possible with encryption. Therefore, to retrieve specific data from the stored data, the cloud provider must be able to search encrypted documents. One solution for this problem is the use of searchable encryption schemes, which enable users to securely execute the search operation on remote data stored on a third party's server. In a cloud computing service, PHI is shared among public users. Thus, searchable encryption should enable multi-users to search for PHI. Therefore, the construction of efficient multi-user searchable encryption is important and challenging.

In the literature, two categories of searchable encryption schemes can be considered: the symmetric key and the public key. Song et al. [6] proposed the first scheme, which enables searchability in symmetric encryption, whereas Boneh et al. [7] introduced the second scheme, which enables public key encryption with keyword search (PEKS). For a network with too many users, the PEKS option is more adaptive than symmetric key searchable encryption.

For example, assume that Bob is a patient who wishes to send his PHI to his doctor, Alice. Bob can establish keywords for his PHI and use Alice's public key to encrypt his PHI and keywords; the encrypted data are sent in the following form:

$$E_{A_{pub}}(PHI) \,||\, PEKS(A_{pub}, kw_1) \,||\cdots||\, PEKS(A_{pub}, kw_n) \qquad (1)$$

where A_{pub} is Alice's public key and $kw_1, kw_2, \cdots, kw_n$ are the keywords that Bob establishes. When Alice wants to search the encrypted documents with the keyword W, she generates a *trapdoor* containing W and sends it to the server. The server locates the corresponding encrypted documents by comparing the trapdoor and the keyword ciphertexts and returns them to Alice. Although the initial security model of PEKS cannot achieve this result because the user can only use one keyword to search the encrypted documents, the ability to use more than one keyword to search a large amount of data can reduce the search scope and improve the query performance. Therefore, Golle et al. [8] proposed secret key encryption with a conjunctive field keyword search scheme in 2004. These authors assumed that m documents and n keyword fields are associated with each document. In 2005, Park et al. [9] presented a conjunctive keyword search scheme based on bilinear paring in the public key cryptosystem; it is named the public key encryption with conjunctive field keyword search (PKCKS). Zhang et al. [10] constructed a public key encryption with conjunctive-subset keywords search algorithm that enables users to list keywords in any order. Sun et al. [11, 12] presented the first verifiable conjunctive keyword search schemes for static and dynamic databases. The verifiable mechanism ensures that the search results are correct and complete.

The limitation in the schemes presented above is that they may require a secure channel between the receiver and the server. Heavy computational and communication loads, such as a Secure Sockets Layer (SSL) between the server and the receiver, are typically required to establish a secure channel. To address this problem, Baek et al. [13] considered removing the secure channel and proposed public key encryption with a keyword search scheme, which is a secure channel-free PEKS (SCF-PEKS). The basic concept of a SCF-PEKS scheme is that the server maintains its public and private key pairs. Whenever the data owner generates the keyword ciphertexts, he inputs the server's public key in the algorithm, and only the corresponding private key can execute the *Test* algorithm in the SCF-PEKS scheme. Rhee et al. [14] strengthened the security model [13] of Baek. However, these schemes can only achieve security in the random oracle model. In another study [15], the secure channel was removed, and the heavy computational and communication loads required to establish a secure channel were resolved.

Without the protection of a secure channel, Byun et al. [16] indicated that the scheme proposed by Baek may be attacked by off-line keyword guessing attacks. Because keywords are chosen from a substantially smaller space than passwords, users usually include well-known keywords for searching documents. Even if the keywords have been encrypted, the attackers can learn the embedded keywords by performing off-line keyword guessing attacks. Inspired by the work of Byun et al. [16], Yau et al. [17] presented an off-line keyword guessing attack on the SCF-PEKS [13] and PKE/PEKS [18] schemes. Rhee et al. [19] constructed a new secure SCF-PEKS scheme against keyword guessing attacks in the random oracle model. Unfortunately, a proof in the random oracle model can only serve as a heuristic argument, and because of the use of contrived constructions, it may generate unsecure schemes when the random oracles are implemented in the standard model [20]. Hwang et al. [21] proposed an efficient secure channel-free public key encryption with a conjunctive field keyword search scheme that can secure against off-line keyword-guessing attacks. Guo et al. [22] proposed an efficient secure channel-free public key encryption with a keyword search scheme that has been shown to be secure in the standard model. We demonstrate that our SCF-PEKS scheme is secure not only against chosen keyword and ciphertext attacks but also against keyword-guessing attacks. Yang et al. [23] proposed a scheme that supports the conjunctive keyword search and resists keyword-guessing attacks. However, this SCF-PECKS scheme uses a significantly complex assumption called the Decisional q-Augmented Bilinear Diffie-Hellman Exponent (q-ABDHE). Miao et al. [24] proposed a significantly more effective and secure cryptographic primitive, called a verifiable conjunctive keyword search over encrypted data without secure channel scheme, to secure against an outside keyword-guessing attack. The combination of artificial intelligence algorithms and file search algorithms can also improve the efficiency of the search algorithm [25] and represents a key research direction.

These scenarios specified a search user; thus, only a single user can search the ciphertext. In the PHR system, the patient Bob uses surgeon Alice's public key to encrypt his PHI and keywords, which enables Alice to search Bob's PHI. However, when pharmacist Lucy must fill a prescription for Bob, she cannot search Bob's PHI. A familiar approach is that Bob encrypts his PHI and keywords again with Lucy's public key; however, this method is inefficient and consumes a significant amount of system storage resources. Therefore, a multi-user search is critical for PHR systems. To solve this problem, Hwang et al. [26] designed a PECKS based on a bilinear map and extended their scheme to a multi-user system, which is the first security model for multi-user public key encryption with a conjunctive keyword search (mPECKS) scheme that requires a secure channel.

Based on this analysis, the conjunctive keyword search scheme for PHRs should achieve the following objectives: remove the secure channel, secure against keyword-guessing attacks, and employ simple assumptions without a random oracle and multi-user conjunctive keyword search. According to these objectives, we propose a novel multi-user conjunctive keyword search scheme (mNCKS) without a secure channel for PHRs, which is referred to as a secure channel-free mNCKS (SCF-mNCKS). This scheme can secure against keyword-guessing attacks. By referencing the access structure [27] of ciphertext policy attribute-based encryption (CP-ABE) [28], the patient constructs the search policy using the keywords of the data files to encrypt the data files and then uploads the ciphertext to the servers. The keyword set L is used to search the data files, and an attribute authority then generates a trapdoor for the users. Users send the trapdoor to the server. If the ciphertext can be successfully tested, the users should obtain the desired data; otherwise, the search fails. A detailed description of this process is provided in Section 3.1.

We propose an efficient mechanism for removing the secure channel. A patient's smart phone is connected to the cloud server provider (CSP) via an unsecure communication channel, such as a GPRS network. The basic concept is for the server to maintain its private and public key pairs. To create a ciphertext, the data owner uses the public keys of the server and the receiver. The receiver can then send a trapdoor to retrieve data associated with the keyword list and send it via a public channel. After receiving the trapdoor, the server can test whether the provided ciphertexts match the trapdoor using its private key. The security of this scheme is verified based on the Decisional Bilinear Diffie-Hellman (DBDH) and Decision Linear (D-Linear) assumptions in the standard model. The scheme comparison shows that the SCF-mNCKS scheme has advantages with respect to security and other functions.

The remainder of this paper is organized as follows: In Section 2, definitions are provided, and we describe the system and security model. In Section 3, we propose the concrete SCF-mNCKS scheme and analyse the security of the proposed generic construction. In Section 4, we describe the performance comparison. In Section 5, we present the conclusions and describe future work.

2. Definitions

In this section, we review the bilinear maps and the computational hardness assumptions and provide the definitions of the proposed construction and selective game model.

2.1 Bilinear Maps

We present a few facts concerning groups with efficiently computable bilinear maps. Assume the efficient algorithm $\prod$ for generating bilinear groups. Through input of a security parameter k, the algorithm $\prod$ outputs a tuple: $\mathbb{G}=[p,G,G_T,g\in G,e]$, where G and G_T are multiplicative groups of prime order p; g is a generator of G; and e is the bilinear map $e:G\times G\rightarrow G_T$. The parameter $e(g,g)$ is the generator of G_T, and Z_p represents a group of large prime order p. The bilinear map e has the following properties:

- Bilinearity: For all $x,y\in G$ and $a,b\in Z_p$, $e(x^a,y^b)=e(x,y)^{ab}$;
- Non-degeneracy: $e(g,g)\neq 1$, where 1 is the identity element of G_T.

We state that G is a bilinear group if the group operation in G and the bilinear map $e:G\times G\rightarrow G_T$ are efficiently computable.

2.2 Complexity Assumptions

Definition 1: DBDH Assumption. *Let $a,b,c,z\in Z_p^*$ be chosen at random, and let g be a generator of G. The DBDH problem in G and G_T is a problem, and the tuple $\{g,g^a,g^b,g^c,Z\}$ should be input to determine whether $Z=e(g,g)^{abc}$. The algorithm $\mathcal{H}$ has the advantage ε in solving the DBDH problem in G and G_T if*

$$Adv_{\mathcal{H}}^{DBDH}(\lambda):=|\Pr[\mathcal{H}(g,g^a,g^b,g^c,e(g,g)^{abc})=1]-\Pr[\mathcal{H}(g,g^a,g^b,g^c,e(g,g)^z)=1]|\geq\varepsilon(\lambda) \quad (2)$$

where $e(g,g)^z\in G_T\setminus\{e(g,g)^{abc}\}$ and λ is the security parameter. We state that the DBDH assumption holds in G if no probabilistic polynomial time (PPT) algorithm has an advantage of at least ε in solving the DBDH problem in G.

Definition 2: D-Linear Assumption. *Let $z_1,z_2,z_3,z_4,z\in Z_p^*$ be chosen at random, and let $g\in G$ be a generator. The D-Linear problem in G is a problem, and the tuple $\{g,g^{z_1},g^{z_2},g^{z_1z_3},g^{z_2z_4},Z\}$ should be input to determine whether $Z=g^{z_3+z_4}$. The algorithm $\mathcal{H}$ has the advantage ε in solving the D-Linear problem in G if*

$$Adv_{\mathcal{H}}^{D\text{-}Linear}(\lambda):=|\Pr[\mathcal{H}(g,g^{z_1},g^{z_2},g^{z_1z_3},g^{z_2z_4},g^{z_3+z_4})=1]-\Pr[\mathcal{H}(g,g^{z_1},g^{z_2},g^{z_1z_3},g^{z_2z_4},g^z)=1]|\geq\varepsilon(\lambda) \quad (3)$$

where $g^z\in G\setminus\{g^{z_3+z_4}\}$ and λ is the security parameter. We state that the D-Linear assumption holds in G, if no PPT algorithm has an advantage of at least ε in solving the D-Linear problem in G.

2.3 Definition of SCF-mNCKS

SCF-mNCKS contains six polynomial time algorithms: $GlobalSetup$, $AASetup$, $KeyGen_{CSP}$, $EncIndex$, $Trapdoor$ and $Test$. These algorithms are presented as follows:

$\boldsymbol{GlobalSetup(1^\lambda)}$: The algorithm is executed by the trusted authority centre (AC) and it takes the security parameter λ as input. The AC generates the global parameter $\mathcal{GP}$. Specifically, $\mathcal{GP}\leftarrow GlobalSetup(1^\lambda)$.

$\boldsymbol{AASetup(\mathcal{GP},U)}$: The algorithm takes the global parameter $\mathcal{GP}$ and keyword set U as inputs. The algorithm returns PK and MSK as the attribute authority's (AA) public key and master private key, respectively. Specifically, $<PK,MSK>\leftarrow AASetup(\mathcal{GP},U)$.

$\boldsymbol{KeyGen_{CSP}(\mathcal{GP})}$: The algorithm takes the global parameter $\mathcal{GP}$ as input and returns pk_{CSP} and sk_{CSP} as the cloud service provider's (CSP) public key and private key, respectively. Specifically, $< pk_{CSP}, sk_{CSP} > \leftarrow KeyGen_{CSP}(\mathcal{GP})$.

$\boldsymbol{EncIndex(\mathcal{GP}, PK, pk_{CSP}, W)}$: The algorithm takes the global parameter $\mathcal{GP}$, AA's public key PK, a CSP's public key pk_{CSP} and the search policy W based on keywords as inputs. The algorithm returns the ciphertext CT. Specifically, $CT \leftarrow EncIndex(\mathcal{GP}, PK, pk_{CSP}, W)$.

$\boldsymbol{Trapdoor(\mathcal{GP}, pk_{CSP}, MSK, L)}$: The algorithm takes the global parameter $\mathcal{GP}$, a CSP's public key pk_{CSP}, AA's master private key MSK and the keyword list L as inputs and outputs a trapdoor TD_L. Specifically, $TD_L \leftarrow Trapdoor(\mathcal{GP}, pk_{CSP}, MSK, L)$.

$\boldsymbol{Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)}$: The algorithm takes the global parameter $\mathcal{GP}$, the CSP's private key sk_{CSP}, the ciphertext CT and the trapdoor TD_L as inputs and determines whether L satisfies W. If L satisfies W, then $"Correct" \leftarrow Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)$.

2.4 Game-Based Security Model

Computational consistency was introduced in [29] and represents a crucial security requirement. To discuss the definitions, we describe the following experiment with the adversary $\mathcal{A}$.

Definition 3: Consistency. *Assume that adversary $\mathcal{A}$ wants to cause a failure in consistency. Consistency is formally defined as follows:*

Setup: *The simulator $\mathcal{B}$ executes $GlobalSetup(1^{\lambda})$, $AASetup(\mathcal{GP}, U)$ and $KeyGen_{CSP}(\mathcal{GP})$.*

Phase 1: *$\mathcal{A}$ submits a keyword list L and a search policy W based on keywords, where $L \mid\neq W$. Then, $EncIndex(\mathcal{GP}, PK, pk_{CSP}, W)$ and $Trapdoor(\mathcal{GP}, pk_{CSP}, MSK, L)$ are executed.*

Challenge: *$Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)$ is executed, where $L \mid\neq W$.*

Guess: *If $"Correct" \leftarrow Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)$, then $\mathcal{A}$ wins the game.*

The advantage of $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}^{cons}(\lambda) := \Pr["Correct" \leftarrow Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)]$. The SCF-mNCKS scheme is computationally consistent if the advantage for all polynomial time adversaries $\mathcal{A}$ to win in this experiment is negligible.

Compared with traditional conjunctive keyword searches, our scheme is based on CP-ABE; therefore, the security model must be redefined. According to the definition of the CP-ABE security model and the characteristics of our scheme, this paper presents a new security game model for our conjunctive keyword search. We formally define the game-based security of SCF-mNCKS, which we refer to as "indistinguishability of secure channel free against chosen keyword attack (IND-CF-CKA)". Informally, IND-CF-CKA guarantees that the CSP, which has not obtained the trapdoors for given keywords, cannot distinguish the correspondence of the ciphertext and keywords. The outside attacker that has not obtained the CSP's private key cannot make any decisions concerning the ciphertexts even if they obtain all trapdoors for the keywords that it holds. Note that the attack models for these two types of attackers are described as $Game_{CSP}$ and $Game_{OA}$. In the SCF-mNCKS security game, the following scenario will occur; this definition is formalized according to a security game between any PPT adversary and a simulator.

Definition 4: IND-CF-CKA. *Let λ be the security parameter and $\mathcal{A}$ be the adversary. We consider the following two games between $\mathcal{A}$ and the simulator $\mathcal{B}$:*

$Game_{CSP}$: $\mathcal{A}$ is assumed to be a CSP.

Init: *The adversary $\mathcal{A}$ sends the challenge search policies W_0 and W_1 based on the keywords to the simulator.*

Setup: *The simulator executes $\mathcal{GP} \leftarrow GlobalSetup(1^{\lambda})$, $< PK, MSK > \leftarrow AASetup(\mathcal{GP}, U)$ and $< pk_{CSP}, sk_{CSP} > \leftarrow KeyGen_{CSP}(\mathcal{GP})$ to obtain the global parameter $\mathcal{GP}$. The public key and master private key pairs (PK, MSK) of AA are calculated. The public and private key pairs (pk_{CSP}, sk_{CSP}) of CSP are set. Then, $\mathcal{B}$ provides (pk_{CSP}, sk_{CSP}) and PK to $\mathcal{A}$.*

Phase 1: *$\mathcal{A}$ submits a keyword list L in a trapdoor query $TD_L = Trapdoor(\mathcal{GP}, pk_{CSP}, MSK, L)$, where $L \not\models W_0 \wedge L \not\models W_1$. The simulator answers with a trapdoor for the keyword list L . Note that these queries can be adaptively repeated.*

Challenge: *The simulator chooses $w \in \{0,1\}$ and executes $CT \leftarrow EncIndex(\mathcal{GP}, PK, pk_{CSP}, W_w)$. The simulator provides the ciphertext CT to $\mathcal{A}$.*

Phase 2: *Same as Phase 1. $\mathcal{A}$ sends L' to the simulator for a query. The simulator answers with a trapdoor for the keyword list. Notice that $L' \not\models W_0 \wedge L' \not\models W_1$.*

Guess: *$\mathcal{A}$ outputs guess $w' \in \{0,1\}$. $\mathcal{A}$ wins if $w' = w$.*

The advantage of $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}^{Game_{CSP}}(\lambda) := |\Pr(w' = w) - 1/2|$.

$Game_{OA}$: $\mathcal{A}$ is assumed to be an outside attacker (including the receiver).

Setup: *The simulator executes $\mathcal{GP} \leftarrow GlobalSetup(1^{\lambda})$, $< PK, MSK > \leftarrow AASetup(\mathcal{GP}, U)$ and $< pk_{CSP}, sk_{CSP} > \leftarrow KeyGen_{CSP}(\mathcal{GP})$ to obtain the global parameter $\mathcal{GP}$. The public key and master private key pairs (PK, MSK) of AA are calculated. The public and private key pairs (pk_{CSP}, sk_{CSP}) of CSP are set. Then, $\mathcal{B}$ provides PK and pk_{CSP} to $\mathcal{A}$.*

Phase 1: *$\mathcal{A}$ submits a keyword list L in a trapdoor query $TD_L = Trapdoor(\mathcal{GP}, pk_{CSP}, MSK, L)$. The simulator answers with a trapdoor for the keyword list L . Note that these queries can be adaptively repeated.*

Challenge: *$\mathcal{A}$ sends the two challenge search policies W_0 and W_1 based on the keywords to the simulator. The simulator chooses $w \in \{0,1\}$ and executes $CT \leftarrow EncIndex(\mathcal{GP}, PK, pk_{CSP}, W_w)$. The simulator provides the ciphertext CT to $\mathcal{A}$.*

Phase 2: *Same as Phase 1. $\mathcal{A}$ sends L' to the simulator for a query. The simulator answers with a trapdoor for the keyword list. In contrast to $Game_{CSP}$, $L' \models W_0 \vee L' \models W_1$ is allowed to be queried as a trapdoor query.*

Guess: *$\mathcal{A}$ outputs guess $w' \in \{0,1\}$. $\mathcal{A}$ wins if $w' = w$.*

The advantage of $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}^{Game_{OA}}(\lambda) := |\Pr(w' = w) - 1/2|$.

Our SCF-mNCKS scheme is said to be IND-CF-CKA secure if $Adv_{\mathcal{A}}^{Game_i}(\lambda)$ is negligible, where i is either CSP or OA .

In the following section, we define the notion of indistinguishability of SCF-mNCKS against a keyword-guessing attack (IND-KGA). Specifically, IND-KGA ensures that an outside adversary (neither the server nor the receiver) that has obtained the trapdoor for a challenge keyword cannot observe the relationship between the trapdoor and any keywords.

Definition 5: Off-Line Keyword-Guessing Attacks on SCF-mNCKS.

Because a trapdoor is sent without a secure channel, an outside adversary is capable of capturing the trapdoor and performing off-line keyword-guessing attacks. The attacker may reveal the encrypted keyword list L that is used by the receiver to search for a document. Similarly, an inside adversary (malicious server) can perform the attack to reveal the keyword in the *trapdoor* and execute the *Test* algorithm to determine the ciphertext that contains the keyword list. However, the outside adversary is unable to distinguish ciphertexts from encrypting a specific keyword list because the *Test* phase requires the server's private key.

A SCF-PEKS scheme that is secure against keyword-guessing attacks, where the attacker is the server, cannot be constructed [30]. Therefore, in this work, we do not consider the keyword-guessing attacks of an inside adversary.

3. SCF-mNCKS Scheme

3.1 Search Policy for Ciphertext

In the CP-ABE scheme, an encryptor specifies an access structure for a ciphertext, which is referred to as an access structure. If a decryptor has the secret key whose associated set of attributes satisfies the access structure, he or she can decrypt the ciphertext.

The access structure and the attribute set that are associated with the secret key in [27] are as follows: Let us assume that the set of attributes in universe $U=\{att_1, att_2, \cdots, att_n\}$ contains n attributes. Each attribute att_i can take two values: 1 and 0. Assume that $L=[L_1, L_2, \cdots, L_n]$ is a set of attributes for a user, and it is called the attribute list. AA generates a secret key for the user based on the user's attribute list. Assume that $W=[W_1, W_2, \cdots, W_n]$ is an access policy to specify the access structure for a ciphertext. Formally, the attribute list $L=[L_1, L_2, \cdots, L_n]$ for a user and the access policy $W=[W_1, W_2, \cdots, W_n]$ for a ciphertext are given. For all i where $1 \le i \le n$, if $L_i = W_i$ or $W_i = *$, L satisfies W, which is represented by the notation $L \models W$. Otherwise, L does not satisfy W, which is represented by the notation $L \not\models W$. The wildcard $*$ can be used in the ciphertext policies and represents the function of a "do not care" value, which can be considered as an AND-gate on all the attributes. For instance, we can let $W=[W_1, W_2, \cdots, W_n]=[0,*,1,*,1,0]$, where $n=6$. If a user has the attribute list $L=[0,1,1,0,1,0]$, he can obtain a secret key associated with $[0,1,1,0,1,0]$ and decrypt the ciphertext encrypted with $[0,*,1,*,1,0]$ but not if the secret key is associated with $[0,1,1,0,1,1]$.

Compared with the access structure of [27], we let $U=\{kw_1, kw_2, \cdots, kw_n\}$ represent the set of keywords of data files that replace the attributes. W is the search policy based on the keywords. For all i where $1 \le i \le n$, each keyword kw_i can take two or more values. More formally, assume that $S_i=\{v_{i,1}, v_{i,2}, \cdots, v_{i,n_i}\}$ is the set of all possible values for kw_i, where n_i is the number of the possible values for kw_i, specifically $n_i = |S_i|$. When the encryptor specifies a wildcard $*$ for W_i, this action corresponds to specifying $W_i = S_i$. We achieve keyword privacy by hiding the subset W_i for each keyword kw_i that is specified in the access structure of the AND-gate of all keywords.

For a PHR system, assume $U=\{kw_1, kw_2, kw_3, kw_4, kw_5\}=\{name, sex, medicalhistory, examination, sensitiveinfo\}$, $n=5$; $S_1=\{v_{1,1}, v_{1,2}, v_{1,3}\}=\{Zhang, Wang, Li\}$, $n_1=3$; $S_2=\{v_{2,1}, v_{2,2}\}=\{male, female\}$, $n_2=2$; $S_3=\{v_{3,1}, v_{3,2}, v_{3,3}\}=\{conditions, allergies, prescriptions\}$, $n_3=3$; $S_4=\{v_{4,1}, v_{4,2}, v_{4,3}, v_{4,4}\}=$

$\{pulse, heart,\ blood\ test,\ X\text{-}ray\ images\}$, $n_4 = 4$; and $S_5 = \{v_{5,1}, v_{5,2}\} = \{HIV, none\}$, $n_5 = 2$. We assume that the search policy W is $Zhang \wedge (male \vee female) \wedge allergies \wedge (pulse \vee heart) \wedge none$ and use W to generate the index. When receivers use L=[$Zhang, male, allergies, heart, none$] to the search data file M , they can obtain the correct result. Receivers cannot obtain the correct result when they use L=[$Li, male, allergies, heart, HIV$] to search the data file.

3.2 Concrete Scheme

The six algorithms are as follows:

GlobalSetup$(\mathbf{1}^{\lambda})$**:** The algorithm is executed by a trusted authority centre and based on the security parameter λ. The authority centre generates the tuple $\mathbb{G} = [p, G, G_T, g \in G, e]$ where g is the generator of G . The global parameter is $\mathcal{GP} = (p, G, G_T, g, e)$.

AASetup$(\mathcal{GP}, U)$**:** Uniformly and randomly choose $y \in Z_p^*$ and compute $Y = e(g, g)^y$. For each keyword $kw_i \in U$, where $1 \le i \le n$, AA chooses random values $\{a_{i,t}, b_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ and random points $\{A_{i,t} \in G\}_{1\le t\le n_i}$. Return $PK = (Y, A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}})$ and $MSK = (y, a_{i,t}, b_{i,t})$ as the AA's public key and master private key, respectively, where $1 \le i \le n$ and $1 \le t \le n_i$.

KeyGen$_{\boldsymbol{CSP}}(\mathcal{GP})$**:** Uniformly and randomly choose $\beta \in Z_p^*$ and compute $B = g^{\beta}$. Return $pk_{CSP} = (\mathcal{GP}, B)$ and $sk_{CSP} = (\mathcal{GP}, \beta)$ as the CSP's public key and private key, respectively.

EncIndex$(\mathcal{GP}, PK, pk_{CSP}, W)$**:** Choose a search policy based on the keywords $W = [W_1, \cdots, W_n]$. The data owner selects the random value $s \in Z_p^*$ and computes $C = Y^s = e(g, g)^{ys}$ and $C_0 = B^s = g^{\beta s}$. For $1 \le i \le n$, the data owner selects random values $\{s_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ and computes $\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_i}$ as follows: if $v_{i,t} \in W_i$, the data owner sets $[C_{i,t,1}, C_{i,t,2}] = [(A_{i,t}^{b_{i,t}})^{s_{i,t}}, (A_{i,t}^{a_{i,t}})^{s-s_{i,t}}]$ (*well-formed*); if $v_{i,t} \notin W$ and $[C_{i,t,1}, C_{i,t,2}]$ are random (*malformed*), then the keyword ciphertext is $CT = (C, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_i}\}_{1\le i\le n})$.

Trapdoor$(\mathcal{GP}, pk_{CSP}, MSK, L)$**:** The user that uses $L = [L_1, \cdots, L_n] = [v_{1,t_1}, \cdots, v_{n,t_n}]$ for searching can obtain the corresponding secret key, which is regarded as the searching trapdoor. For $1 \le i \le n$, AA selects random values $r_i, \lambda_i \in Z_p^*$, sets $r = \sum_{i=1}^n r_i$, and computes $D_0 = g^{(y-r)}$. Then, $D_{i,0} = g^{r_i} (A_{i,t_i})^{a_{i,t_i} b_{i,t_i} \lambda_i}$, $D_{i,1} = B^{a_{i,t_i} \lambda_i}$ and $D_{i,2} = B^{b_{i,t_i} \lambda_i}$ are computed by AA. The searching trapdoor is $TD_L = (D_0, \{D_{i,0}, D_{i,1}, D_{i,2}\}_{1\le i\le n})$.

Test$(\mathcal{GP}, sk_{CSP}, CT, TD_L)$**:** The data user sends the searching trapdoor TD_L to the CSP to implement the search request. The CSP executes the algorithm to verify whether the data user that corresponds to the keyword list L satisfies the search policy W . If L satisfies W , then CSP computes $C_0' = C_0^{1/\beta}$. For $1 \le i \le n$, we let $[C_{i,1}', C_{i,2}'] = [C_{i,t_i,1}, C_{i,t_i,2}]$, where $L_i = v_{i,t_i}$. If equation (4) holds, then return "Correct"; otherwise, return "Incorrect".

$$C\left(\prod_{i=1}^n e(C_{i,1}', D_{i,1}) \cdot e(C_{i,2}', D_{i,2})\right)^{1/\beta} \overset{?}{=} e(C_0', D_0) \prod_{i=1}^n e(C_0', D_{i,0}) \tag{4}$$

Correctness. In the following, we show that a correctly generated ciphertext can be correctly tested by the CSP who has the correct trapdoor. Let the ciphertext $CT = (C, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_1}\}_{1\le i\le n})$, which is associated with search policy $W = [W_1, W_2, \cdots, W_n]$, be based on keywords. The trapdoor is $TD_L = (D_0, \{D_{i,0}, D_{i,1}, D_{i,2}\}_{1\le i\le n})$. This process produces the following equation:

$$\begin{aligned}&\frac{C\left(\prod_{i=1}^{n} e(C'_{i,1}, D_{i,1})e(C'_{i,2}, D_{i,2})\right)^{1/\beta}}{e(C'_0, D_0)\prod_{i=1}^{n} e(C'_0, D_{i,0})} = \frac{C\prod_{i=1}^{n} e((A_{i,t}^{b_{i,t}})^{s_{i,t}}, g^{\beta a_{i,t_i}\lambda_i})^{1/\beta} e((A_{i,t}^{a_{i,t}})^{s-s_{i,t}}, g^{\beta b_{i,t_i}\lambda_i})^{1/\beta}}{e(C'_0, D_0)\prod_{i=1}^{n} e(g^s, g^{r_i}(A_{i,t_i})^{a_{i,t_i}b_{i,t_i}\lambda_i})} \\ &= \frac{e(g,g)^{ys}}{e(g^s, g^{(y-r)})\prod_{i=1}^{n} e(g^s, g^{r_i})} = \frac{e(g,g)^{ys}}{e(g,g)^{ys-sr}e(g,g)^{sr}} = 1\end{aligned} \tag{5}$$

$$C\left(\prod_{i=1}^{n} e(C'_{i,1}, D_{i,1})\cdot e(C'_{i,2}, D_{i,2})\right)^{1/\beta} = e(C'_0, D_0)\prod_{i=1}^{n} e(C'_0, D_{i,0}) \tag{6}$$

3.3 Security Analysis

Theorem 1. Our SCF-mNCKS scheme is computationally consistent.

Proof: Assume that a polynomial-time adversary $\mathcal{A}$ can attack the computational consistency of our scheme. Let (W, L) denote the search policy based on keywords and the keyword list for a user, which $\mathcal{A}$ returns in the consistency experiment. Without a loss of generality, assume that L does not satisfy W.

Select a random value $s \in Z_p^*$. Let $C = Y^s = e(g,g)^{ys}$ and $C_0 = B^s = g^{\beta s}$. For $1 \le i \le n$, the data owner picks random values $\{s_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ and computes $\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_i}$ as follows: if $v_{i,t} \in W_i$, then the data owner sets are $[C_{i,t,1}, C_{i,t,2}] = [(A_{i,t}^{b_{i,t}})^{s_{i,t}}, (A_{i,t}^{a_{i,t}})^{s-s_{i,t}}]$ (*well-formed*); if $v_{i,t} \notin W_i$, then $[C_{i,t,1}, C_{i,t,2}]$ are random (*malformed*). Let $TD_L = (D_0, \{D_{i,0}, D_{i,1}, D_{i,2}\}_{1\le i\le n})$ where $D_0 = g^{(y-r)}$, $D_{i,0} = g^{r_i}(A_{i,t_i})^{a_{i,t_i}b_{i,t_i}\lambda_i}$, $D_{i,1} = B^{a_{i,t_i}\lambda_i}$ and $D_{i,2} = B^{b_{i,t_i}\lambda_i}$. For $1 \le i \le n$, AA selects random values $r_i, \lambda_i \in Z_p^*$ and sets $r = \sum_{i=1}^{n} r_i$. Note that $\mathcal{A}$ wins exactly when L does not satisfy W if formula (7) is established.

$$e(C'_0, D_0)\prod_{i=1}^{n} e(C'_0, D_{i,0}) = e(g,g)^{ys}\left(\prod_{i=1}^{n} e(C'_{i,1}, D_{i,1})e(C'_{i,2}, D_{i,2})\right)^{1/\beta} \tag{7}$$

where $[C'_{i,1}, C'_{i,2}] = [C_{i,t_i,1}, C_{i,t_i,2}]$.

Assume that $L_k \nvDash W_k$, then $[C'_{k,1}, C'_{k,2}] = [g^{z_1}, g^{z_2}]$.

$$\begin{aligned}&e(C'_0, D_0)\prod_{i=1}^{n} e(C'_0, D_{i,0}) = e(g,g)^{ys}\left(\prod_{i=1}^{n} e(C'_{i,1}, D_{i,1})e(C'_{i,2}, D_{i,2})\right)^{1/\beta} \\ &\Leftrightarrow e(C'_0, D_0)\prod_{i=1}^{n} e(C'_0, D_{i,0}) = e(g,g)^{ys}\prod_{i=1}^{n} e(C'_{i,1}, D_{i,1}^{1/\beta})e(C'_{i,2}, D_{i,2}^{1/\beta}) \\ &\Leftrightarrow \prod_{i=1}^{n} e(g,g)^{sr_i}\prod_{i=1}^{n} e(g^s, (A_{i,t_i})^{a_{i,t_i}b_{i,t_i}\lambda_i}) = e(g,g)^{sr}\prod_{i=1, i\ne k}^{n} e(A_{i,t_i}, g)^{a_{i,t_i}b_{i,t_i}\lambda_i s} e(g^{z_1}, g^{a_{k,t_k}\lambda_k})e(g^{z_2}, g^{b_{k,t_k}\lambda_k}) \\ &\Leftrightarrow e(g, A_{k.t_k})^{a_{k,t_k}b_{k,t_k}\lambda_k s} = e(g^{z_1}, g^{a_{k,t_k}\lambda_k})e(g^{z_2}, g^{b_{k,t_k}\lambda_k}) \\ &\Leftrightarrow e(g, A_{k.t_k})^{a_{k,t_k}b_{k,t_k}\lambda_k s} = e(g^{z_1 z_2 a_{k,t_k}b_{k,t_k}}, g^{\lambda_k}) \\ &\Leftrightarrow e(g^{a_{k,t_k}b_{k,t_k}\lambda_k}, A_{k,t_k}^s) = e(g^{a_{k,t_k}b_{k,t_k}\lambda_k}, g^{z_1 z_2}) \\ &\Leftrightarrow e(g^{a_{k,t_k}b_{k,t_k}\lambda_k}, g^{z_k s}) = e(g^{a_{k,t_k}b_{k,t_k}\lambda_k}, g^{z_1 z_2})\end{aligned} \tag{8}$$

Because z_1 and z_2 are random, z_k and s are kept secret from the receiver. Therefore, $Pr[z_1 z_2 = z_k s] = 1/(p^2 - p)$, where $p-1$ is the total element number in Z_p^*. As previously described, when $L \nvDash W$ and $"Correct" \leftarrow Test(\mathcal{GP}, sk_{CSP}, CT, TD_L)$, the following result is obtained:

$$Adv_{\mathcal{A}}^{cons}(\lambda) = Pr[z_1 z_2 = z_k s] \le 1/(p^2 - p) \tag{9}$$

We previously discussed the security analysis of the SCF-mNCKS with a single authority approach and provided proof using the DBDH and D-Linear assumptions. The analysis of $Game_{CSP}$ and $Game_{OA}$ is as follows:

Theorem 2: SCF-mNCKS satisfies the indistinguishability of keywords using the DBDH and D-Linear assumptions in $Game_{CSP}$.

Assume that adversary $\mathcal{A}$ commits to the challenge search policies W_0 and W_1 at the beginning of the game. We employ the notation $W_w = [W_{w,1}, W_{w,2}, \cdots, W_{w,n}]$. The proof uses a sequence of hybrid games to argue that adversary $\mathcal{A}$ cannot win the original security game denoted by G_0 with non-negligible probability. We begin by slightly modifying game G_0 into game G_1. Games G_0 and G_1 are identical, with the exception of how the challenge ciphertext is generated. In G_1, if adversary $\mathcal{A}$ did not obtain the TD_L, whose associated keyword list L is such that $L \nvDash W_0 \wedge L \nvDash W_1$, then the challenge ciphertext component CT is a random element of G_T, regardless of the random coin u. The remainder of the ciphertext is generated as usual. If the adversary obtained the TD_L whose associated keyword list L is such that $L \vDash W_0 \wedge L \vDash W_1$, then the challenge ciphertext in G_1 is generated correctly ($G_0 = G_1$ in this case).

Next, we modify G_1 by changing the manner of generating the components $\{\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_1}\}_{1 \le i \le n}$ and define a sequence of the games as follows: If $v_{i,t}$ exists such that $(v_{i,t} \in W_{0,i} \wedge v_{i,t} \notin W_{1,i})$ or $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \in W_{1,i})$, then the components $\{C_{i,t,1}, C_{i,t,2}\}$ that are properly generated in game G_{l-1} are replaced with the random values in the new modified game G_1 regardless of the random coin u. We define a new game when this replacement occurs. We repeat this replacement one by one until no components satisfy $(v_{i,t} \in W_{0,i} \wedge v_{i,t} \notin W_{1,i})$ or $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \in W_{1,i})$. In the last game of the sequence, the advantage of the adversary is zero because the adversary is given a ciphertext that is chosen from the same distribution regardless of the random coin u. Thus, we obtain the indistinguishability of ciphertext by constantly modifying the games.

Lemma 1: The advantage of distinguishing game G_0 and game G_1 is negligible for any PPT adversary $\mathcal{A}$ using the DBDH assumption.

Proof: Given the DBDH challenge tuple $[g, g^a, g^b, g^c, Z]$ by the simulator, where Z is either $(g,g)^{abc}$ or random with equal probability. We let $v \in \{0,1\}$. If $v = 0$, then $Z = e(g,g)^{abc}$; if $v = 1$, then $Z = e(g,g)^z$. Assume that the PPT adversary $\mathcal{A}$ exists who distinguishes game G_0 and game G_1 with the non-negligible advantage ε. Thus, we can construct the simulator $\mathcal{B}$ that will break the DBDH assumption with the advantage $\varepsilon / 2$. Then, the simulator $\mathcal{B}$ creates the following simulation:

Init: $\mathcal{A}$ gives $\mathcal{B}$ the two challenge search policies $W_0 = [W_{0,1}, \cdots, W_{0,n}]$ and $W_1 = [W_{1,1}, \cdots, W_{1,n}]$ based on the keywords.

Setup: Let λ be the security parameter and $\mathcal{GP} = (p, G, G_T, g, e)$ be the global parameter. $\mathcal{B}$ selects $w \in \{0,1\}$ and sets $W_w = [W_{w,1}, W_{w,2}, \cdots, W_{w,n}]$. $\mathcal{B}$ selects $\delta \in Z_p^*$ and computes $B = g^{\delta}$. Establish $pk_{CSP} = (\mathcal{GP}, B)$ and $sk_{CSP} = (\mathcal{GP}, \delta)$ as the CSP's public key and private key, respectively. Compute $Y = e(g^a, g^b) = e(g,g)^{ab}$, which implies $y = ab$. For each keyword kw_i where $1 \le i \le n$, $\mathcal{B}$ randomly chooses $\{\alpha_{i,t} \in Z_p^*\}_{1\le t\le n_i}$. If $v_{i,t} \notin W_{w,i}$, then $\mathcal{B}$ generates $A_{i,t} = g^{a\alpha_{i,t}}$; if $v_{i,t} \in W_{w,i}$, then $\mathcal{B}$ generates $A_{i,t} = g^{\alpha_{i,t}}$. $\mathcal{B}$ selects $\{a_{i,t}, b_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ at random for each keyword kw_i. Let $PK = (Y, A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}})$ and $MSK = (y, a_{i,t}, b_{i,t})$ represent AA's public key and master private key, respectively, where $1 \le i \le n$ and $1 \le t \le n_i$. Then, $\mathcal{B}$ provides (PK, pk_{CSP}, sk_{CSP}) to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ submits the keyword list $L = [L_1, \cdots, L_n] = [v_{1,t_1}, \cdots, v_{n,t_n}]$ in a trapdoor query. We only consider the case where $L \nvDash W_0 \wedge L \nvDash W_1$. When $L \vDash W_0 \vee L \vDash W_1$, $\mathcal{B}$ simply aborts and takes a random guess.

When $L \nvDash W_0 \wedge L \nvDash W_1$, $k \in \{1, 2, \cdots, n\}$ must exist such that $L_k = v_{k,t_k} \notin W_{w,k}$. For $1 \le i \le n$, $\mathcal{B}$ selects $r_i' \in Z_p^*$ at random. $\mathcal{B}$ then sets $r_k = ab + r_i'$. For every $i \ne k$, $\mathcal{B}$ sets $r_i = r_i'$. $\mathcal{B}$ sets $r = \sum_{i=1}^n r_i = ab + \sum_{i=1}^n r_i'$. D_0 can be computed as $D_0 = g^{y-r} = g^{ab-r} = g^{-\sum_{i=1}^n r_i'}$.

For k, $[D_{k,0}, D_{k,1}, D_{k,2}] = [g^{r_k}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k}, B^{a_{k,t_k}\lambda_k}, B^{b_{k,t_k}\lambda_k}]$ is computed as follows:

$$D_{k,0} = g^{r_k}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} = g^{ab+r_k'}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} = g^{ab+r_k'}(g^{a\alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda_k} = g^{r_k'}(g^{a\alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda_k'} \quad (10)$$

where λ_k is chosen at random such that $\lambda_k = \lambda_k' - b/(\alpha_{k,t_k} a_{k,t_k} b_{k,t_k})$ and random λ_k' is known by $\mathcal{B}$. $\mathcal{B}$ can easily compute the components $[D_{k,1}, D_{k,2}]$. For $i \ne k$, $\mathcal{B}$ can also easily compute $[D_{i,0}, D_{i,1}, D_{i,2}]$. Therefore, all valid components of TD_L have been constructed.

Challenge: The simulator $\mathcal{B}$ calculates $C = Z$ and $C_0 = B^c = g^{\delta c}$. When $v_{i,t} \in W_{w,i}$, $\mathcal{B}$ can correctly generate $\{C_{i,t,1}, C_{i,t,2}\}$ because $A_{i,t}$ does not contain an unknown a. When $v_{i,t} \notin W_{w,i}$, $\{C_{i,t,1}, C_{i,t,2}\}$ can be chosen at random, and then $\mathcal{B}$ sends $CT = (C, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_1}\}_{1\le i\le n})$ to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ continues issuing queries to oracle *Trapdoor* to receive the trapdoor of keyword list L such that $L \nvDash W_w$, where $w \in \{0,1\}$.

Guess: $\mathcal{A}$ outputs the guess $w' \in \{0,1\}$. $\mathcal{B}$ outputs 0 if $w' = w$ or 1 if $w' \ne w$. Two cases are observed as follows:

• If $Z = e(g,g)^{abc}$, $v = 0$. Then, $\mathcal{A}$ is in the original game G_0. The advantage of $\mathcal{A}$ is ε; thus, $Pr[w' = w | v = 0] = 1/2 + \varepsilon$. When $w' = w$, $\mathcal{B}$ outputs $v' = 0$. We have $Pr[v' = v | v = 0] = 1/2 + \varepsilon$.

• If $Z = e(g,g)^z$, $v = 1$. Then, $\mathcal{A}$ is in the modified game G_1. $\mathcal{A}$ has no advantage to distinguish bit w; thus, $Pr[w' \ne w | v = 1] = 1/2$. When $w' \ne w$, $\mathcal{B}$ outputs $v' = 1$. We have $Pr[v' = v | v = 1] = 1/2$.

Therefore, we know that the advantage of $\mathcal{B}$ in this DBDH game is as follows:

$$\begin{aligned} Pr[v' = v] - 1/2 &= Pr[v' = v | v = 1]Pr[v = 1] + Pr[v' = v | v = 0]Pr[v = 0] - 1/2 \\ &= \tfrac{1}{2}\cdot\tfrac{1}{2} + (\tfrac{1}{2} + \varepsilon)\cdot\tfrac{1}{2} - \tfrac{1}{2} = \tfrac{\varepsilon}{2} \end{aligned} \quad (11)$$

Lemma 2: The advantage of distinguishing game G_{l-1} and game G_l is negligible for any PPT adversary $\mathcal{A}$ using the D-Linear assumption.

Proof: Given a D-Linear challenge $[g, g^{z_1}, g^{z_2}, g^{z_2 z_4}, g^{z_3+z_4}, Z]$ by the simulator, where Z is either $e(g,g)^{z_1 z_3}$ or random with equal probability. Let $v \in \{0,1\}$. If $v = 0$, then $Z = e(g,g)^{z_1 z_3}$;

otherwise, Z is random. We assume that $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \in W_{1,i})$ without a loss of generality. Assume that the PPT adversary $\mathcal{A}$ exists who distinguishes the game G_{l-1} and game G_l with the non-negligible advantage ε. We build an algorithm $\mathcal{B}$ that can simulate the game with the advantage $\varepsilon/2$. Then, the simulator $\mathcal{B}$ creates the following simulation.

Init: The simulator $\mathcal{B}$ executes $\mathcal{A}$. $\mathcal{A}$ provides $\mathcal{B}$ two challenge search policies based on the keywords $W_0 = [W_{0,1}, W_{0,2}, \cdots, W_{0,n}]$ and $W_1 = [W_{1,1}, W_{1,2}, \cdots, W_{1,n}]$. Then, $\mathcal{B}$ flips a random coin $w \in \{0,1\}$. If $w = 0$, we have $G_{l-1} = G_l$, and the advantage of $\mathcal{A}$ between G_{l-1} and G_l is equivalent. Thus, we consider the case where $w = 1$.

Setup: Let λ be the security parameter and $\mathcal{GP} = (p, G, G_T, g, e)$ be the global parameter. $\mathcal{B}$ selects $w \in \{0,1\}$ and sets $W_w = [W_{w,1}, W_{w,2}, \cdots, W_{w,n}]$. $\mathcal{B}$ selects $\delta \in Z_p^*$ and computes $B = g^{\delta}$. Let $pk_{CSP} = (\mathcal{GP}, B)$ and $sk_{CSP} = (\mathcal{GP}, \delta)$ represent the CSP's public key and private key, respectively. Compute $Y = e(g,g)^y$ where y is known to $\mathcal{B}$. For each keyword kw_i where $1 \le i \le n$, $\mathcal{B}$ chooses $\{\alpha_{i,t} \in Z_p^*\}_{1 \le t \le n_i}$ at random. If $v_{i,t} \notin W_{w,i}$, then $\mathcal{B}$ generates $A_{i,t} = g^{a\alpha_{i,t}}$; if $v_{i,t} \in W_{w,i}$, then $\mathcal{B}$ generates $A_{i,t} = g^{\alpha_{i,t}}$. $\mathcal{B}$ randomly selects $\{a_{i,t}, b_{i,t} \in Z_p^*\}_{1 \le t \le n_i}$ for each keyword kw_i. For a_{i_l,t_l} and b_{i_l,t_l}, $\mathcal{B}$ sets $a_{i_l,t_l} = z_1$ and $b_{i_l,t_l} = z_2$. Then, $\mathcal{B}$ computes $A_{i_l,t_l}^{a_{i_l,t_l}} = g^{\alpha_{i_l,t_l} a_{i_l,t_l}}$ and $A_{i_l,t_l}^{b_{i_l,t_l}} = g^{\alpha_{i_l,t_l} b_{i_l,t_l}}$ without knowing z_1 and z_2. Let $PK = (Y, A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}})$ and $MSK = (y, a_{i,t}, b_{i,t})$ represent AA's public key and master private key, respectively, where $1 \le i \le n$ and $1 \le t \le n_i$. Then, $\mathcal{B}$ provides (PK, pk_{CSP}, sk_{CSP}) to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ submits the keyword list $L = [L_1, L_2, \cdots, L_n] = [v_{1,t_1}, v_{2,t_2}, \cdots, v_{n,t_n}]$ in a trapdoor query. If $L_{i_l} \ne v_{i_l,t_l}$, then $\mathcal{B}$ can easily generate the corresponding trapdoor. Let us assume that $L_{i_l} = v_{i_l,t_l}$. $\mathcal{B}$ must compute the trapdoor components $[D_{i_l,0}, D_{i_l,1}, D_{i_l,2}] = [g^{r_{i_l}} (A_{i_l,t_l})^{a_{i_l,t_l} b_{i_l,t_l} \lambda_{i_l}}, B^{a_{i_l,t_l} \lambda_{i_l}}, B^{b_{i_l,t_l} \lambda_{i_l}}]$, where $a_{i_l,t_l} = z_1$ and $b_{i_l,t_l} = z_2$.

$\mathcal{B}$ can compute $D_{i_l,0}$ as follows: $D_{i_l,0} = g^{r_{i_l}} (A_{i_l,t_l})^{a_{i_l,t_l} b_{i_l,t_l} \lambda_{i_l}} = g^{r_{i_l}} (A_{i_l,t_l})^{z_1 z_2 \lambda_{i_l}} = g^{r_{i_l}} (g^{\alpha_{i_l,t_l}})^{z_1 z_2 \lambda_{i_l}} = g^{r'_{i_l}}$, where r_{i_l} is randomly chosen such that $r_{i_l} = r'_{i_l} - \alpha_{i_l,t_l} z_1 z_2 \lambda_{i_l}$ and random r'_{i_l} is known by $\mathcal{B}$. $\mathcal{B}$ can easily compute the components $[D_{i_l,1}, D_{i_l,2}]$ without knowing z_1 and z_2.

Here, we can assume $L \nvDash W_0 \wedge L \nvDash W_1$ because $L_{i_l} = v_{i_l,t_l} \wedge v_{i_l,t_l} \notin W_{1-w,i_l}$, that is, the result is $L \nvDash W_{1-w}$; therefore, $L \nvDash W_w$. Thus, $k \in \{1, \cdots, n\}$ such that $L_k = v_{k,t_k} \notin W_{w,k}$. Then, $\mathcal{B}$ generates $[D_{k,0}, D_{k,1}, D_{k,2}]$ as follows: $\mathcal{B}$ sets $r_k = r'_k + \alpha_{i_l,t_l} z_1 z_2 \lambda_{i_l}$, where r'_k is random and known by $\mathcal{B}$, and computes $D_{k,0} = g^{r_k} (A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} = g^{r'_k + \alpha_{i_l,t_l} z_1 z_2 \lambda_{i_l}} (g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda_k} = g^{r'_k} (g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda'_k}$, where λ_k is randomly chosen such that $\lambda_k = \lambda'_k - (\alpha_{i_l,t_l} z_2 \lambda_{i_l}) / (\alpha_{k,t_k} a_{k,t_k} b_{k,t_k})$. Random λ'_k is known by $\mathcal{B}$. $\mathcal{B}$ can easily compute the components $[D_{k,1}, D_{k,2}]$ without knowing z_2. For $i \ne i_l$, $\mathcal{B}$ can also easily compute $[D_{i,0}, D_{i,1}, D_{i,2}]$.

Finally, by calculating

$$r = \sum_{i=1}^{n} r_i = r_{i_l} + r_k + \sum_{i \ne i_l, k} r_i = r'_{i_l} - \alpha_{i_l,t_l} z_1 z_2 \lambda_{i_l} + r'_k + \alpha_{i_l,t_l} z_1 z_2 \lambda_{i_l} + \sum_{i \ne i_l, k} r_i = r'_{i_l} + r'_k + \sum_{i \ne i_l, k} r_i \tag{12}$$

the component $D_0 = g^{y-r}$ of the trapdoor can be computed. Therefore, all valid components of

TD_L have been constructed.

Challenge: $\mathcal{B}$ sets $C_0 = B^{z_3+z_4} = g^{\delta(z_3+z_4)}$, which implies $s = z_3 + z_4$. If $L \nvDash W_0 \wedge L \nvDash W_1$ for every queried L, then $\mathcal{B}$ sets C to be random; otherwise, $C = e(g, g^{z_3+z_4})^y$. $\mathcal{B}$ generates the components $\{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_1}\}_{1\le i\le n}$ for W_w as for G_{l-1}, with the exception that $\{C_{i_l,t_l,1}, C_{i_l,t_l,2}\}$ are computed as $C_{i_l,t_l,1} = (A_{i_l,t_l}^{b_{i_l,t_l}})^{s_{i_l,t_l}} = (A_{i_l,t_l}^{z_2})^{z_4} = (g^{\alpha_{i_l,t_l} z_2})^{z_4}$ and $C_{i_l,t_l,2} = (A_{i_l,t_l}^{a_{i_l,t_l}})^{s-s_{i_l,t_l}} = (g^{\alpha_{i_l,t_l} z_1})^{z_3} = Z^{\alpha_{i_l,t_l}}$ without knowing $z_2 z_4$ and $z_1 z_3$, which implies that $s_{i_l,t_l} = z_4$ and $Z = g^{z_1 z_3}$. If $Z = g^{z_1 z_3}$, then the components are *well formed*. Then, $\mathcal{A}$ is in game G_{l-1}. $CT = (C, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_1}\}_{1\le i\le n})$ is sent to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ continues issuing queries to the oracle *Trapdoor* to receive the trapdoor of the keyword list L such that $L \nvDash W_w$, where $w \in \{0,1\}$.

Guess: $\mathcal{A}$ outputs the guess $w' \in \{0,1\}$. $\mathcal{B}$ outputs 0 if $w' = w$ or 1 if $w' \ne w$. Two cases are observed as follows:

• If $Z = e(g,g)^{z_1 z_3}$, $v = 0$. Then, $\mathcal{A}$ is in the modified game G_{l-1}. The advantage of $\mathcal{A}$ is ε; thus, $Pr[w' = w | v = 0] = 1/2 + \varepsilon$. When $w' = w$, $\mathcal{B}$ outputs $v' = 0$. We have $Pr[v' = v | v = 0] = 1/2 + \varepsilon$.

• If $Z = e(g,g)^z$, $v = 1$. Then, $\mathcal{A}$ is in game G_l, and $\mathcal{A}$ has no advantage to distinguish bit w; thus, $Pr[w' \ne w | v = 1] = 1/2$. When $w' \ne w$, $\mathcal{B}$ outputs $v' = 1$. We have $Pr[v' = v | v = 1] = 1/2$.

Therefore, we know that the advantage of $\mathcal{B}$ in this D-Linear game is as follows:

$$\begin{aligned} Pr[v' = v] - 1/2 &= Pr[v' = v | v = 1]Pr[v = 1] + Pr[v' = v | v = 0]Pr[v = 0] - 1/2 \\ &= \tfrac{1}{2}\cdot\tfrac{1}{2} + (\tfrac{1}{2} + \varepsilon)\cdot\tfrac{1}{2} - \tfrac{1}{2} = \tfrac{\varepsilon}{2} \end{aligned} \tag{13}$$

By considering the sequence of the games ($G_0, G_1, \cdots$) starting with the original game G_0, no polynomial adversary can win G_0 with a non-negligible advantage based on these lemmas.

Theorem 3: SCF-mNCKS satisfies the indistinguishability of the keywords in the DBDH assumption in $Game_{OA}$.

Proof: Given a DBDH challenge tuple $[g, g^a, g^b, g^c, Z]$ by the simulator, where Z is either $e(g,g)^{abc}$ or random with equal probability. Let $v \in \{0,1\}$. If $v = 0$, then $Z = e(g,g)^{abc}$; otherwise, $Z = e(g,g)^z$. Assume that the adversary $\mathcal{A}$ in $Game_{OA}$ wins the selective game with the advantage ε. Thus, we can construct the simulator $\mathcal{B}$ that will break the DBDH assumption with an advantage over ε. Then, the simulator $\mathcal{B}$ creates the following simulation.

Setup: Let λ be the security parameter and $\mathcal{GP} = (p, G, G_T, g, e)$ be the global parameter. Let $B = g^c$ and the CSP's public key be $pk_{CSP} = (\mathcal{GP}, B)$. Compute $Y = e(g^a, g^b) = e(g,g)^{ab}$, which implies that $y = ab$. For each $kw_i \in U$ where $1 \le i \le n$, $\mathcal{B}$ chooses random values $\{a_{i,t}, b_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ and random points $\{A_{i,t} \in G\}_{1\le t\le n_i}$. Return $PK = (Y, A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}})$ and $MSK = (y, a_{i,t}, b_{i,t})$ as the AA's public key and master private key, respectively, where $1 \le i \le n$ and $1 \le t \le n_i$. Then, $\mathcal{B}$ provides (pk_{CSP}, PK) to $\mathcal{A}$.

Phase 1: If $\mathcal{A}$ submits the keyword list $L = [L_1, L_2, \cdots, L_n] = [v_{1,t_1}, v_{2,t_2}, \cdots, v_{n,t_n}]$ in a trapdoor query, then $\mathcal{B}$ computes $TD_L = (D_0, \{D_{i,0}, D_{i,1}, D_{i,2}\}_{1\le i\le n})$ as usual and returns it as the answer.

Challenge: $\mathcal{A}$ submits the two challenge policies $W_0 = [W_{0,1}, \cdots, W_{0,n}]$ and $W_1 = [W_{1,1}, \cdots, W_{1,n}]$ based on the keywords. $\mathcal{B}$ chooses $w \in \{0,1\}$ and calculates $C = Z$. $\mathcal{B}$ selects the random value $s \in Z_p^*$ and calculates $C_0 = B^s = g^{sc}$. For $1 \le i \le n$, $\mathcal{B}$ randomly selects $\{s_{i,t} \in Z_p^*\}_{1\le t\le n_i}$ and computes $\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_i}$ as follows: if $v_{i,t} \in W_{w,i}$, then the data owner sets $[C_{i,t,1}, C_{i,t,2}] = [(A_{i,t}^{b_{i,t}})^{s_{i,t}}, (A_{i,t}^{a_{i,t}})^{s-s_{i,t}}]$ (*well-formed*); if $v_{i,t} \notin W_{w,i}$, then $[C_{i,t,1}, C_{i,t,2}]$ are random (*malformed*). $\mathcal{B}$ sends $CT = (C, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1\le t\le n_i}\}_{1\le i\le n})$ to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ continues issuing queries to the oracle *Trapdoor* to receive the trapdoor of the keyword list L. Note that $L \vDash W_w$ is allowed.

Guess: $\mathcal{A}$ outputs the guess $w' \in \{0,1\}$. $\mathcal{B}$ outputs 0 if $w' = w$ or outputs 1 if $w' \ne w$. Two cases are observed as follows:

- If $Z = e(g,g)^{abc}$, then $\mathcal{A}$ must satisfy $|Pr[w' = w] - 1/2| \ge \varepsilon$;
- If $Z = e(g,g)^z$, then $\mathcal{A}$ has no advantage to distinguish the bit w. Specifically, $Pr[w' = w] = 1/2$.

Therefore, we can obtain equation (14), which completes the proof of $Game_{OA}$.

$$|Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] - Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^z) = 1]| \ge |(1/2 \pm \varepsilon) - 1/2| = \varepsilon \tag{14}$$

Theorem 4: SCF-mNCKS is secure against outside keyword-guessing attacks.

Proof: Assume that the outside adversary $\mathcal{A}$ exists who can eavesdrop on the trapdoor TD_L. $\mathcal{A}$ can obtain the global parameter $\mathcal{GP} = (p, G, G_T, g, e)$, AA's public key $PK = (Y, A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}})$ and CSP's public key $pk_{CSP} = (\mathcal{GP}, B)$ from the public network. To obtain the encrypted keywords, $\mathcal{A}$ first guesses the keyword list L and then executes the keyword guessing attacks as follows:

$$\begin{aligned}
&\prod e(A_{i,t}^{a_{i,t}}, D_{i,2}) \overset{?}{=} Ye(g, D_0) \prod e(g, D_{i,0})\\
&\prod e(A_{i,t}^{a_{i,t}}, B^{b_{i,t_i}\lambda_i}) \overset{?}{=} e(g,g)^y e(g, g^{y-r}) \prod e(g, g^{r_i} (A_{i,t_i})^{a_{i,t_i} b_{i,t_i} \lambda_i})\\
&\prod e(A_{i,t}^{a_{i,t}}, g^{\beta b_{i,t_i}\lambda_i}) \overset{?}{=} e(g,g)^{-r} \prod e(g,g)^{r_i} e(g, (A_{i,t_i}^{a_{i,t_i}})^{b_{i,t_i}\lambda_i})\\
&\prod e(g, (A_{i,t}^{a_{i,t}})^{\beta b_{i,t_i}\lambda_i}) \overset{?}{=} \prod e(g, (A_{i,t_i}^{a_{i,t_i}})^{b_{i,t_i}\lambda_i})
\end{aligned} \tag{15}$$

In these derivations, AA's private key β is unknown to $\mathcal{A}$; therefore, the adversary $\mathcal{A}$ cannot successfully attack our scheme by performing keyword-guessing attacks.

4. Performance Comparison

We analysed our scheme by comparing it with the schemes of [10, 21, 23] in terms of the features, storage overhead and computation efficiency. Let $|p|$ be the size of the elements in Z_p, and let $|g|$ and $|g_T|$ be the element size in G and G_T, respectively ($|g_1|$ and $|g_2|$ are denoted by $|g|$). Let n denote the total number of keywords in the system, let n_i denote the number of the values of each keyword, and let n_c denote the number of keywords that are associated with the ciphertext. G_e denotes the operation of the exponentiation of an elliptic curve, and P denotes the bilinear pairing operations.

4.1 Feature Comparison

Table 1 shows a comparison of the features of certain aspects of the system. These schemes and our scheme are based on the prime order of bilinear groups. Yang's scheme [23] and our scheme are based on symmetric bilinear groups and have advantages over Zhang's scheme [10] and Hwang's scheme [21], which are based on asymmetric bilinear groups. Zhang's scheme [10] and Yang's scheme [23] are based on the strong assumption of q-BDHEA. Our scheme and Hwang's scheme [21] are based on a simple assumption. Zhang [10] did not provide any security proof; therefore, we do not know whether Zhang [10] requires a random oracle. Only our scheme supports a multi-user search, which is suitable for PHRs.

Table 1. Features Comparison

Features	**Zhang [10]**	**Hwang [21]**	**Yang [23]**	**Ours**
Bilinear Maps	Asymmetric	Asymmetric	Symmetric	Symmetric
Assumption	p-DDHI	DDH	q-ABDHE, DBDH	DBDH, DL
Random Oracle	-	Without	Without	Without
SCF	No	Yes	Yes	Yes
KGA	No	Yes	Yes	Yes
Multi-user CKS	No	No	No	Yes

4.2 Storage Overhead

Table 2 shows the comparison of the storage overhead on each entity in the system. To achieve the multi-user search, our scheme requires the AA to generate a master private key, which generates a trapdoor for each receiver. The main storage overhead on the AA is generated by the master key. In our scheme, the AA must generate a master key for each keyword value. All public parameters contribute to the storage overhead on the owner. We do not consider the storage overhead caused by the encrypted data because the storage overhead is the same in our scheme and the other schemes. In our scheme and schemes [21, 23], the CSP is required to store the ciphertext and a private key. The storage overhead on each receiver in the four scenarios is associated with the number of keywords that they possess, whereas other schemes require each receiver to store an additional private key.

Table 2 shows that the storage overhead of our scheme is similar to the storage overhead of the other schemes. Only the owner's storage overhead is larger than that of the other schemes. However, the keyword space is generally small; therefore, the total gap is not significant. Our approach is verified using simple assumptions and supports multi-user searches.

Table 2. Comparison of Storage Overhead

Schemes	**AA (MSK)**	**Owner (All PK)**	**CSP (CT)**	**Receiver (Trapdoor & SK)**
Zhang [10]	-	$O(n)\|g\|$	$O(n)\|g\|+O(1)\|p\|$	$O(n)\|g\|+O(1)\|p\|$
Hwang [21]	-	$O(1)\|g\|+O(1)\|g_T\|$	$O(n)\|g\|+O(1)\|p\|$	$O(n)\|g\|+O(1)\|p\|$
Yang [23]	-	$O(1)\|g\|$	$O(n)\|g\|+O(1)\|g_T\|+O(1)\|p\|$	$O(n)\|g\|+O(1)\|p\|$
Ours	$O(n)\|p\|$	$O(n)\|g\|+O(1)\|g_T\|$	$O(n)\|g\|+O(1)\|g_T\|+O(1)\|p\|$	$O(n)\|g\|$

4.3 Computation Efficiency

Table 3 shows a comparison of the execution time of *Encryption*, *Trapdoor* and *Test*. Our scheme can achieve a richer conjunctive keyword search, as demonstrated in Section 3.1. For convenience, we allow our conjunctive keyword search to be identical to other schemes, that is, $n_i = 1$ and $n_c = n$. The proposed scheme is similar to the other three scenarios in terms of the execution time of *Encryption*, *Trapdoor* and *Test*. The time complexity of the *Trapdoor* and *Test* in the Hwang [21] scheme is a constant order; although it requires the multiplication of an elliptic curve of N times in practice. Because the time required for the multiplication operation is significantly smaller than the exponential operation, the multiplication time is omitted in **Table 3**. In the actual simulation process, the execution time of *Trapdoor* and *Test* increases with an increase in the number of keywords.

Table 3. Comparison of Computation Efficiency

Schemes	Encryption Time	Trapdoor Time	Test Time
Zhang [10]	$O(n)G_e + O(1)P$	$O(n)G_e$	$O(n)P$
Hwang [21]	$O(n)G_e$	$O(1)G_e$	$O(1)G_e + O(1)P$
Yang [23]	$O(n)G_e$	$O(n)G_e$	$O(n)P$
Ours	$O(n)G_e + O(1)P$	$O(n)G_e$	$O(n)P$

Experiment Setup: We conducted the experiment on a 64-bit Ubuntu 14.04 operating system with an Intel CoreTM i5-6200U (2.3 GHz) processor and 8 G RAM. The experimental code uses the Pairing-based Cryptography Library (PBC-0.5.14) and cpabe-0.11 to implement the schemes. We employ 160-bit elliptic curve groups in the hyper-singular curves $y^2 = x^3 + x$ based on 512-bit finite fields. The simulation experiment system is built, and the operation time is tested in the system. Specifically, the pairing operation time of the PBC library is approximately 5.5 ms, and the exponential operation times of G and G_T are approximately 5.1 ms and 0.9 ms, respectively. In the simulation process, the relationship between the number of keywords and the execution time of *Encryption*, *Trapdoor* and *Test* is tested. We select a 1 MB file and encrypt the file with a different number of keywords. We test the execution time of the *Encryption*, *Trapdoor* and *Test* processes as the number of keywords changes (from 1 to 20). All simulation results are the mean of 30 trials. According to this method, the other three schemes are simulated. The relationships are shown in **Fig. 1**.

Fig. 1 shows a comparison of the computational efficiency of the four schemes with different values of n, which is the number of keywords. The horizontal axis represents the number of keywords in the search, and the vertical axis represents the execution time of *Encryption*, *Trapdoor* and *Test*.

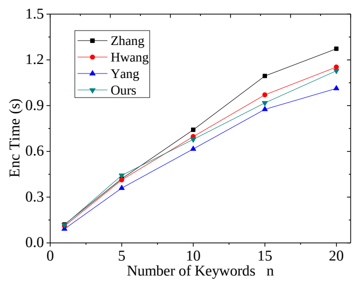

Fig. 1(a). Encryption

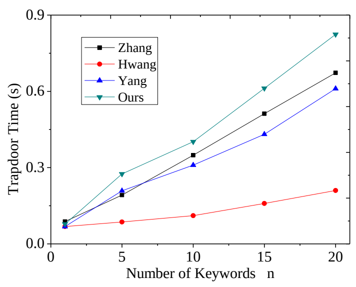

Fig. 1(b). Trapdoor

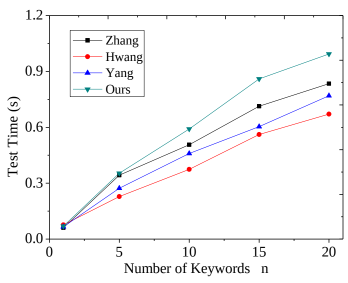

Fig. 1(c). Test

Fig. 1(a) plots the encryption time of the algorithms executed by the data owner. The encryption time increases with the number of keywords. When the number of keywords changes from 1 to 20, the encryption time of each scheme increases approximately linearly. The encryption time of all schemes is equivalent, which is consistent with the theoretical analysis in **Table 3**.

Fig. 1(b) shows the execution time of *Trapdoor*, which is executed by the receiver (*Trapdoor* is executed by the AA in our scheme). The schemes of **[10, 23]** and our scheme have the same time complexity; therefore, the three curves in the graph are similar. The execution time of Trapdoor is $G_e + 2G_m + nH$ in Hwang **[21]**, where G_m denotes the multiplication of an elliptic curve and H denotes the operation of a hash (in the theoretical analysis, the time for the hash operation and the multiplication operation is omitted for simplicity). Because the calculation of H is small, the slope of Hwang **[21]** is very small, that is, with an increase in the number of keywords, the execution time of Trapdoor in **[21]** changes slightly less than the other three schemes. This finding is consistent with the theoretical analysis in **Table 3**. Our scheme has the longest execution time, and *Trapdoor* is executed by the AA in our scenario.

Fig. 1(c) shows the execution time of *Test*, which is executed by the CSP. The execution time of *Test* is positively correlated with the number of keywords. In the actual simulation process, the execution time of Test is $5G_e + nG_m + 5P$ in Hwang **[21]**. Because the multiplication is much faster than the bilinear pairing operations, the execution time of the multiplication is omitted in the theoretical analysis. Thus, in the case of the same number of keywords, the scheme in **[21]** requires a shorter execution time, which is consistent with the theoretical analysis in **Table 3**. The execution time of our scheme is longer than that of the other schemes. However, *Test* is performed by the CSP, which possesses strong computing power. In practical applications, this gap can be ignored.

However, our scheme achieves a multi-user conjunctive keyword search that is suitable for PHRs. Other schemes are not suitable.

5. Conclusions and Future Work

In this paper, we propose the novel multi-user conjunctive keyword search scheme without secure channel against keyword-guessing attacks (SCF-mNCKS) for PHRs. In this scheme, the owner constructs the search policy W based on the data file keywords to encrypt the data files and then uploads the ciphertext to the servers. The keyword list L is employed to search the data files. The scheme achieves the search purpose by determining whether L satisfies W. Because sensitive PHI is highly valuable, the trapdoors that are associated with the PHI keywords are often the targets of malicious behaviour, which may expose the PHI. To ensure trapdoor security while reducing the computational and communication burden, we propose an efficient mechanism for removing the secure channel. Our scheme resists keyword-guessing attacks and achieves a multi-user conjunctive keyword search that is suitable for PHRs. The security of this scheme is verified using the DBDH and D-Linear assumptions in the standard model. The scheme comparison shows that the SCF-mNCKS scheme has advantages regarding security and functions under analogous efficiency.

In future research, we will consider enhancing this scheme to achieve adaptive security. The current security lacks a formal security definition to capture keyword-guessing attacks. Building the formal security definition is a problem that we will continue to study.

References

[1] A. Bhargav-Spantzel, J. Camenisch, T. Gross and D. Sommer, "User centricity: a taxonomy and open issues," *Journal of Computer Security*, vol. 15, no. 5, pp. 493-527, July, 2007. Article (CrossRef Link).

[2] H. Löhr, A. R. Sadeghi and M. Winandy, "Securing the e-health cloud," in *Proc. of the 1st ACM International Health Informatics Symposium*, pp. 220-229, November 11-12, 2010. Article (CrossRef Link).

[3] A. Mehmood, H. Song and J. Lloret, "Multi-agent based framework for secure and reliable communication among open clouds," *Network Protocols and algorithms. Macrothink Institute*, vol. 6, no. 4, pp. 60-76, December, 2014. Article (CrossRef Link).

[4] I. Butun, M. Erol-Kantarci, B. Kantarci and H. Song, "Cloud-centric multi-level authentication as a service for secure public safety device networks," *IEEE Communications Magazine*, vol. 54, no. 4, pp. 47-53, April, 2016. Article (CrossRef Link).

[5] M. Shojafar, S. Abolfazli, H. Mostafaei and M. Singhal, "Improving channel assignment in multi-radio wireless mesh networks with learning automata," *Wireless Personal Communications*, vol. 82, no. 1, pp. 61-80, May, 2015. Article (CrossRef Link).

[6] D. X. Song, D. Wagner and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. of IEEE Symposium on Security and Privacy*, pp. 44-55, May 14-17, 2000. Article (CrossRef Link).

[7] D. Boneh, G. Di Crescenzo, R. Ostrovsky and G. Persiano, "Public key encryption with keyword search," in *Proc. of International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 506-522, May 2-6, 2004. Article (CrossRef Link).

[8] P. Golle, J. Staddon and B. Waters, "Secure conjunctive keyword search over encrypted data," in *Proc. of International Conference on Applied Cryptography and Network Security*, pp. 31-45, June 8-11, 2004. Article (CrossRef Link).

[9] D. J. Park, K. Kim and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *Proc. of International Workshop on Information Security Applications*, pp. 73-86, August 23-25, 2004. Article (CrossRef Link).

[10] B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search," *Journal of Network and Computer Application*, vol. 34, no. 1, pp. 262-267, January, 2011. Article (CrossRef Link).

[11] W. Sun, B. Wang, N. Cao and et al, "Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 11, pp. 3025-3035, November, 2014. Article (CrossRef Link).

[12] W. Sun, X. Liu, W. Lou and et al, "Catch you if you lie to me: Efficient verifiable conjunctive keyword search over large dynamic encrypted cloud data," in *Proc. of IEEE conference on computer communications*, pp. 2110-2118, April 26-May 1, 2015. Article (CrossRef Link).

[13] J. Baek, R. Safavinaini and W. Susilo, "Public Key Encryption with Keyword Search Revisited," in *Proc. of International Conference on Computational Science and Its Applications*, pp. 1249-1259, June 30-July 3, 2008. Article (CrossRef Link).

[14] H. S. Rhee, J. H. Park, W. Susilo and D. H. Lee, "Improved searchable public key encryption with designated tester," in *Proc. of ACM Symposium on Information, Computer and Communications Security*, pp. 376-379, March 10-12, 2009. Article (CrossRef Link).

[15] K. Emura, A. Miyaji, M. S. Rahman and K. Omote, "Generic constructions of secure-channel free searchable encryption with adaptive security," *Security and Communication Networks*, vol. 8, no. 8, pp. 1547-1560, May, 2015. Article (CrossRef Link).

[16] J. W. Byun, H. S. Rhee, H. A. Park and D. H. Lee, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in *Proc. of Workshop on Secure Data Management*, pp. 75-83, September 10-11, 2006. Article (CrossRef Link).

[17] W. C. Yau, S. H. Heng and B. M. Goi, "Off-line keyword guessing attacks on recent public key encryption with keyword search schemes," in *Proc. of International Conference on Autonomic and Trusted Computing*, pp. 100-105, June 23-25, 2008. Article (CrossRef Link).

[18] J. Baek, R. Safavi-Naini and W. Susilo, "On the integration of public key data encryption and public key encryption with keyword search," in *Proc. of International Conference on Information Security*, pp. 217-232, August 30-September 2, 2006. Article (CrossRef Link).

[19] H.S. Rhee, W. Susilo and H-J. Kim, "Secure searchable public key encryption scheme against keyword guessing attacks," *IEICE Electron. Express*, vol. 6, no. 5, pp. 237-243, June, 2009. Article (CrossRef Link).

[20] R. Canetti, O. Goldreich and S. Halevi, "The random oracle methodology, revisited," *Journal of the ACM* (*JACM*), vol. 51, no. 4, pp. 557-594, July, 2004. Article (CrossRef Link).

[21] M. S. Hwang, S. T. Hsu and C. C. Lee, "A new public key encryption with conjunctive field keyword search scheme," *Information Technology And Control*, vol. 43, no .3, pp. 277-288, February, 2014. Article (CrossRef Link).

[22] L. Guo and W. C. Yau, "Efficient secure-channel free public key encryption with keyword search for EMRs in cloud storage," *Journal of medical systems*, vol. 39, no. 2, pp. 1-11, February, 2015. Article (CrossRef Link).

[23] Y. Yang and M. Ma, "Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 746-759, April, 2016. Article (CrossRef Link).

[24] Y. Miao, J. Ma, F. Wei and et al, "VCSE: Verifiable conjunctive keywords search over encrypted data without secure-channel," *Peer-to-Peer Networking and Applications*, pp. 1-13, May, 2016. Article (CrossRef Link).

[25] M. Shojafar, J. H. Abawajy, Z. Delkhah and et al, "An efficient and distributed file search in unstructured peer-to-peer networks," *Peer-to-Peer Networking and Applications*, vol. 8, no. 1, pp. 120-136, January, 2015. Article (CrossRef Link).

[26] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Proc. of International Conference on Pairing-Based Cryptography*, pp. 2-22, July 2-4, 2007. Article (CrossRef Link).

[27] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. of the 14th ACM conference on Computer and communications security*, pp. 456-465, October 29-November 02, 2007. Article (CrossRef Link).

[28] J. Bethencourt, A. Sahai and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. of IEEE symposium on security and privacy*, pp. 321-334, May 20-23, 2007. Article (CrossRef Link).

[29] M. Abdalla, M. Bellare, D. Catalano and et al, "Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions," in *Proc. of Annual International Cryptology Conference*, pp. 205-222, August 14-18, 2005. Article (CrossRef Link).

[30] L. Fang, W. Susilo, C. Ge and J. Wang, "Public key encryption with keyword search secure against keyword guessing attacks without random oracle," *Information Sciences*, vol. 238, no. 7, pp. 221-241, July, 2013. Article (CrossRef Link).

Zhiyuan Zhao received an M.S. degree from Zhengzhou Information Science and Technology Institute, Zhengzhou, China in 2015. He is currently pursuing a Ph.D. degree at Zhengzhou Information Science and Technology Institute, China. His research interests include cryptograph theory, especially attribute-based encryption.

Jianhua Wang received a Ph.D. degree from Zhengzhou Information Science and Technology Institute, China in 2008 and became a Full Professor in 2006. His research interests primarily focus on information security. He has authored and co-authored more than 100 journal and conference papers.