A Generic Construction for Verifiable Attribute-based Keyword Search Schemes

Mohammmad Hassan Ameri, Maryam Rajabzadeh Assar, Javad Mohajeri, Mahmoud Salmasizadeh

Abstract—Cloud data owners encrypt their documents before outsourcing to provide their privacy. They could determine a search control policy and delegate the ability of search token generation to the users whose attributes satisfy the search control policy. Verifiable attribute-based keyword search (VABKS) where the users can also verify the accuracy of cloud functionality is one of such schemes. In this paper, the first generic construction for VABKS is proposed. To this end, the notion of hierarchical identity-based multi-designated verifier signature (HIB-MDVS) has been introduced and existential forgery under chosen message attack (EF-CMA) is formally defined for its unforgeability. Furthermore, anonymity against chosen identity vector set and chosen plaintext attack (Anon-CIVS-CPA) has been defined as the security definition of hierarchical identity-based broadcast encryption (HIBBE) in a formal way. The proposed construction is built in a modular structure by using HIBBE, HIB-MDVS, and Bloom filter as the building blocks. We prove that the security of proposed construction is based on the unforgeability of HIB-MDVS and the anonymity of HIBBE. Finally, the concept of verifiable ranked keyword search will be introduced and a construction of this primitive will be presented which is based on proposed VABKS.

Index Terms—Cloud computing, searchable encryption, keyword ranked search, attribute-based encryption, privacy preserving, hierarchical identity-based cryptography, provable security.

1 Introduction

The need for secure cloud computing and cloud storage systems has made encrypted data potential target of numerous research studies. If a user wishes to outsource its sensitive data in an untrusted cloud, it will be necessary to encrypt them before outsourcing them to the cloud. Therefore, to retrieve special data among the stored data, the cloud provider should be able to search on encrypted documents which cause difficulty in some cases. One solution for this problem is using searchable encryption schemes.

In the literature, two types of searchable encryption schemes can be considered for this primitive which are referred to as public key and symmetric key searchable encryptions. Song et.al. for the first time, proposed symmetric searchable encryption [1], and Boneh et.al. introduced the concept of public key encryption with keyword search (PEKS) [2]. In a symmetric searchable encryption, a user, who is the sole authority capable of generating a search token, uses a secret randomness to generate a searchable ciphertext. In some cases where a network contains too many users, it would be more efficient to use public key searchable encryption because the users know the receivers' public key and could generate a searchable ciphertext such that the entities with the corresponding secret keys be the ones who can generate some valid search tokens. In what follows, the public key searchable encryption will be elaborated on in more detail.

1.1 Public Key Searchable Encryption

After the introduction of the concept of PEKS, this primitive was followed by many researchers and consequently various structures were proposed to improve its security and enhance its efficiency. The existing constructions can be classified in two groups of generic and concrete constructions.

Boneh et.al. applied PEKS to intelligent email routing, defined the indistinguishability under chosen keyword attack (IND-PEKS-CKA) and proved that their proposed concrete construction is IND-PEKS-CKA secure [2]. They also proposed a generic construction of PEKS based on identity-based encryption (IBE) [3]. Also, Abdalla et.al. [4], defined consistency of PEKS and presented a generic construction of anonymous identity-based encryption [5] to PEKS. They also introduced identity-based encryption with keyword search (IBEKS) and public key encryption with temporary keyword search (PETKS) and defined PETKS-IND-CPA security. Finally, they proposed a generic construction of PETKS with PETKS-IND-CPA security from anonymous hierarchical identity-based encryption (HIBE) [4].

The PEKS scheme needs a secure channel between the receiver and the gateway. Thus, Beak et.al. proposed a solution to eliminate the requirements of a secure channel [6]. They defined secure channel free public key encryption with keyword search (SCF-PEKS) and indistinguishability under chosen keyword attack (IND-SCF-CKA) [6].

In the real world, the number of keywords which are used as the auxiliary information for conducting the search operation is limited, the PEKS schemes are susceptible to offline keyword guessing attack. Byun et.al. for the first time in 2006, applied offline keyword guessing attack to PEKS [7]. In 2009, Tang et.al. proposed the concept of public key encryption with registered keyword search (PERKS) [8]. In this scheme, the sender registers both keyword and its corresponding receiver, before generating a search token. Although the process of keyword preregistration makes searchable encryption more complex, it makes the scheme immune to the offline keyword guessing attack.

In 2010, for the first time, Li et.al. introduced the notion of public key fuzzy keyword search over encrypted data [9]. In this variant, to enhance search efficiency and keyword privacy, the cloud returns the documents which contain both the same

- *M. H. Ameri is with the Department of Electrical Engineering, Sharif university of Technology, Tehran, Iran. E-mail: ameri_mohammadhasan@ee.sharif.edu*
- *M. R. Assar, J. Mohajeri and M. Salmasizadeh are with the Electronics Research Institute of Sharif university of Technology, Tehran, Iran. E-mail: asaar@ee.sharif.edu, mohajer@sharif.edu, salmasi@sharif.edu*

and similar keywords to the queried keywords, except for those which contain the exact keywords. In 2012, Nishioka defined perfect keyword privacy (PKP) and IND-PKP as a security definition for PEKS [10]. In 2013, Peng et.al. proposed the notion of public key encryption with fuzzy keyword search (PEFKS) which is secure against offline keyword guessing attack [11]. They defined semantic security under chosen keyword attack (SS-CKA) and indistinguishability of keywords under non-adaptively chosen keywords and keyword guessing attack (IK-NCK-KGA) and presented a generic construction from anonymous identity-based encryption to PEFKS. They proved that the resulting PEFKS is SS-CKA secure. In 2013, Zhang et.al. [12], studied the relation between predicate encryption (PE) [13] and PEKS and they introduced public key encryption with fine grained keyword search (PEFKS). They defined IND-PEFKS-CPA security and designed a generic construction of PE for PEFKS. In 2014 Han et.al. introduced attribute-based encryption with keyword search (ABEKS) and presented a generic construction of key policy attribute-based encryption (KP-ABE) [14] for ABEKS [15].

In 2014, Zheng et.al. proposed the notion of attribute-based keyword search (ABKS) and proposed a concrete construction based on bilinear pairings. In this model, the data owner determines a search control policy and generates a searchable ciphertext according to this policy and sends it to the cloud for storage. Then specific set of data users who satisfies the search control policy are allowed to generate a search token for an arbitrary set of keywords requiring no interactions between the data users and the data owners [16]. In [15] the same scenario will happen whereas the interaction between the data users and data owner is undeniable. The data user sends the generated search token to the cloud to search for the required encrypted data. It may happen that the cloud try to send an invalid search result to the data users. Zheng et.al. also constructed a verifiable ABKS (VABKS) in a modular fashion by using attribute-based encryption, bloom filter, digital signature, and ABKS, to verify the accuracy of search results received from the cloud. They also defined the security of VABKS against the selectively chosen keyword attack (SCKA) and verifiability of VABKS as the security definitions of VABKS.

As IBE and designated multi verifier signature scheme (DMVS) [17], [18] are considered to have a close relationship to ABE [14], we have found that hierarchical identity-based broadcast encryption (HIBBE) [19] and hierarchical identity-based multi-designated verifier signature (HIB-MDVS) could be useful to construct a generic scheme for VABKS.

1.2 Our Contributions

The contribution of the present paper is three-fold. First, we introduce a cryptographic notion called HIB-MDVS, and also present formal definitions for its unforgeability, the existential forgery under chosen-message attack (EF-CMA). We also define anonymity of HIBBE against chosen identity vector set and chosen plaintext attack (Anon-CIVS-CPA) in the standard model. Second, a generic construction for VABKS which is built based on a bloom filter [20], HIBBE [19] and the new cryptographic notion HIB-MDVS. HIB-MDVS is used to establish the search control policy and provide verifiability. We also used HIBBE for generating a searchable ciphertext for a group of legitimate data users. Security of the proposed construction is based on the unforgeability of HIB-MDVS and the anonymity of HIBBE which is formally proved in this paper. Third, introducing the concept of verifiable ranked keyword search and proposing a construction for this primitive which is based on our generic construction of VABKS. The notable feature of our proposed scheme of verifiable ranked keyword search is that it eliminates the interactions between the data owners and users for exchanging the secret key which is used for generating the search tokens and this is important because we can reduce the communication overhead of ranked keyword search schemes. And the other notable feature of our verifiable ranked keyword search scheme is that this scheme is constructed based on our generic construction of VABKS which its security has already been proved.

1.3 Organization

The rest of this paper is organized as follows. Section II reviews using notation and cryptographic primitives. Section III expresses the system and security model. Section IV presents the genetic construction for VABKS based on HIB-MDVS and HIBBE. Section V analyzes the security of the proposed generic construction and in section VI we introduce the concept of verifiable ranked keyword search and present a construction by using the proposed VABKS. Finally, we conclude the paper in section VII.

2 Preliminaries

In an attribute-based encryption [14], the ciphertext is generated for a group of users with some specific attributes according to an access control policy. We use I_{KeyGen} to specify the set of attributes for each user, and I_{Enc}, as the set of attributes which determines the access control policy. Also notation $T(.,.)$ is used here to check whether a set of attributes satisfies an access control policy. If I_{KeyGen} satisfies access control policy I_{Enc} then $T(I_{\mathsf{KeyGen}}, I_{\mathsf{Enc}}) = 1$; otherwise, $T(I_{\mathsf{KeyGen}}, I_{\mathsf{Enc}}) = \perp$. Notation $a \xleftarrow{\$} S$ denotes selecting element a from set S uniformly at random, $|S|$ denotes the cardinality of set S, $\mathbf{ID} = (ID_1, ID_2, \ldots, ID_{d'})$ denotes an identity vector, and $||\mathbf{ID}|| = d'$ denotes the depth of $\mathbf{ID}$. The network has a hierarchical structure like a tree and each user is associated with an identity vector. The existing identities are indexed with a number according to the tree-like hierarchical structure of the network and the identity vector of each user contains two parts: the first part is identity vector of its parents and the second part is user's identity. Figure 1 shows a network with hierarchical structure and the identity vectors of the users of this network in more detail. Denote d as the maximum value of d' which expresses the number of level of hierarchy in the network. $\mathbf{V}$ is defined here as a set of identity vectors, where $||\mathbf{V}|| = \max\{||\mathbf{ID}|| : \mathbf{ID} \in \mathbf{V}\}$ denotes the maximal depth of $\mathbf{V}$. Furthermore, the identity vector set $Par(\mathbf{ID}) = \{(ID_1, ID_2, \ldots, ID_{d''}) : d'' \leq d'\}$ is defined as the parent of the identity vector $\mathbf{ID} = (ID_1, ID_2, \ldots, ID_{d'})$ and consequently $Par(\mathbf{V}) = \bigcup_{\mathbf{ID} \in \mathbf{V}} Par(\mathbf{ID})$ is the parent set of $\mathbf{V}$. Table 1 is a list of notations which are commonly used in this paper.

2.1 Bloom Filter

Let an entity would like to check whether an element is a member of a set and its elements are not known. Bloom filter (BF) - was proposed for the first time in 1970 [20] - can be used to solve this problem. The output of a BF is a m-bit array where

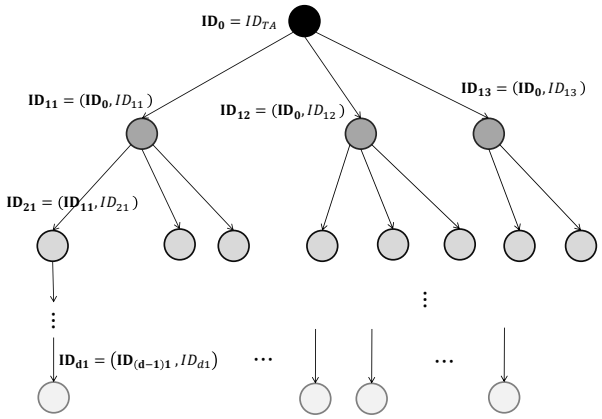

Fig. 1. Hierarchical model of the network. $\mathbf{ID}_{TA}$ expresses the identity of trusted authority and ID_{ij} denotes the identity of jth user in the ith level of hierarchy and $\mathbf{ID}_{ij}$ is the identity vector of the user whose identity is ID_{ij}

TABLE 1
Frequently Used Notations

Symbols	Description
$\mathbf{ID}_{ij}$	The identity vector of each user of network, denotes as $\mathbf{ID} = (ID_1, \ldots, ID_{d'})$
$\mathbf{V}$	Identity vectors set, denotes as $\mathbf{V} = \{\mathbf{ID}_1, \ldots, \mathbf{ID}_{\lvert V\rvert}\}$
$Par(\mathbf{ID})$	The parents of identity vector $\mathbf{ID}$
$Par(\mathbf{V})$	The parent of identity vector set $\mathbf{V}$
$\lvert\lvert\mathbf{ID}\rvert\rvert$	Depth of identity vector $\mathbf{ID}$
$a \xleftarrow{\$} S$	Randomly selection of a from set S
I_{Enc}	Expresses the access control policy in ABE
I_{KeyGen}	Set of attributes of a user
KG	The set of keywords, $KG = \{\omega_1, \ldots, \omega_n\}$
Doc	The set of all files which should be outsourced in encrypted format to the cloud, $Doc = \{F_1, \ldots, F_m\}$
$\mathcal{F}(\omega_i)$	The set of all files which contain keyword ω_i, $\mathcal{F}(\omega_i) = \{F_{ij} : \omega_i \in F_{ij}\}$
$\lvert S\rvert$	Denotes the cardinality of set S
$Out \leftarrow \mathsf{Alg}(In)$	Denotes that algorithm Alg outputs Out on input In
$A := B$	Allocating the value of B to A

all of the m bits are initialized as zero. The BF uses a group of k collision resistant hash functions $H'_1, H'_2, \ldots, H'_k$ with the same ranges, i.e. for $1 \leq i \leq k, H'_i : \{0,1\}^* \rightarrow \{0, \ldots, m-1\}$, and the output of this hash functions determines the index of a bit in the m-bit array (We remind that the index of each bit in the BF is numbered from 0 to $m-1$). Let $S = \{\omega_1, \ldots, \omega_n\}$ and ω be an arbitrary element of S. To compute the output of the BF it should have been computed $ind_j = H'_j(\omega)$ for all $1 \leq j \leq k$ and set the ind_jth bit of m-bit array to 1. When an entity wants to check the existence of ω in S, it should compute $ind_j = H'_j(\omega)$ for all $1 \leq j \leq k$ and check whether ind_jth bit of BF is equal to 1. If for all $1 \leq j \leq k$, ind_jth bit of BF is equal to 1, then ω belongs to the set S with a high probability; otherwise, ω does not belong to S. In what follows, the two algorithms which are used to generate and verify a m-bit BF are expressed [20].

- $\mathsf{BF} \leftarrow \mathsf{BFGen}(\{H'_1, \ldots, H'_k\}, \{\omega_1, \ldots, \omega_n\})$: This algorithm takes k universal hash functions $\{H'_1, \ldots, H'_k\}$ and data set $S = \{\omega_1, \ldots, \omega_n\}$ as input to generate a m-bit BF.
- $\{0,1\} := \mathsf{BFverify}(\{H'_1, \ldots, H'_k\}, \mathsf{BF}, \omega)$: The output of this algorithm is 1 if the element $\omega \in S$, and otherwise the output is 0.

3 System and Security Model

3.1 Verifiable Attribute Based Encryption with Keyword Search (VABKS)

Each VABKS consists of four parties [16]: (1) *Data Owner*, who outsources its encrypted documents to the cloud in a searchable manner, (2) *The Cloud Provider*, who stores a massive amount of data and can search on encrypted data, (3) *Data User*, who obtains its required documents associated with some special keywords which have been encrypted and stored in the cloud by data owner, and (4) *Trusted Authority* (TA), who generates secret keys of users and issues them to the data users and data owners through a secure and authenticated channel. The architecture of VABKS is shown in Figure 2.

Suppose that the data owner wishes to generate a verifiable searchable keyword ciphertext for just one keyword group $KG = \{\omega_1, \ldots .\omega_n\}$. Note that we encrypt the keywords which belong to the keywords group KG to generate searchable ciphertexts. Typically, each VABKS contains six polynomial time algorithms: Setup, KeyGen, BuildIndex, TokenGen, SearchIndex, Verify [16]. These algorithms are presented as follows.

- $(PP, MSK) \leftarrow \mathsf{Setup}(\lambda)$: This algorithm takes security parameter λ as input, and generates master secret key MSK, and public parameter PP.
- $(SK) \leftarrow \mathsf{KeyGen}(MSK, I_{\mathsf{KeyGen}})$: This algorithm takes user's attribute set I_{KeyGen} and master secret key MSK, to generate secret key SK of the user associated to I_{KeyGen}.
- $(D_{\mathsf{cph}}, \mathsf{Index}) \leftarrow \mathsf{BuildIndex}(I_{\mathsf{Enc}}, KG)$: This algorithm takes access control policy I_{Enc} and keyword group KG as inputs. The output of this algorithm is tuple ciphertext $(D_{\mathsf{cph}}, \mathsf{Index})$ where D_{cph} is the searchable encryption of keywords group KG and Index is the auxiliary information. The cloud runs the search operation on D_{cph} and uses Index to prove to the data user that the result of search is correct.
- $(st_\omega) \leftarrow \mathsf{TokenGen}(SK, \omega)$: This algorithm takes secret key SK and keyword ω as inputs and generates search token st_ω as the outputs of this algorithm.
- $\{(\mathsf{proof}, \mathsf{rslt})\} \leftarrow \mathsf{SearchIndex}(st_\omega, \mathsf{Index}, D_{\mathsf{cph}})$: This algorithm takes search token st_ω, tuple ciphertext $(D_{\mathsf{cph}}, \mathsf{Index})$ as inputs and generates the result and proof pair $\{(\mathsf{proof}, \mathsf{rslt})\}$. The cloud provider runs this algorithm and sends result and proof pair $\{(\mathsf{proof}, \mathsf{rslt})\}$ to the data user. proof proves to the data user that the cloud has run the search operation correctly. This algorithm return 1 as the result of search if
  - $T(I_{\mathsf{KeyGen}}, I_{\mathsf{Enc}}) = 1$ & (The attributes set of data user (I_{KeyGen}) satisfies access control policy I_{Enc} which is determined by the data owner.)

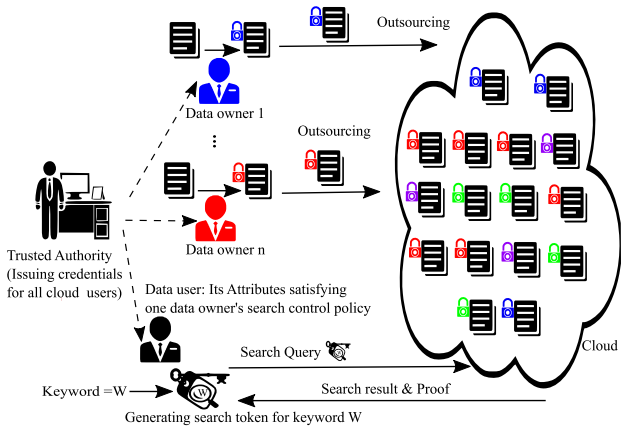

Fig. 2. The VABKS model in the network. The data users encrypt their documents and the data user generates a search token and sends it to the cloud. The cloud uses this token to search for queried documents and returns the search result and the proof of the search.

  - $\omega \in KG$ (Keywords group KG contains the keyword corresponding to search token st_ω.)

- $\{0,1\} := \mathsf{Verify}(SK, \omega, st, \mathsf{proof}, \mathsf{rslt})$: This algorithm takes secret key SK, keyword ω, search token st, result and proof pair $(\mathsf{rslt}, \mathsf{proof})$ as inputs to verify whether the search results are valid or not. This algorithm returns 1 if $(\mathsf{rslt}, \mathsf{proof})$ is valid with reference to search token st; otherwise return 0.

3.1.1 SCKA Security Game for VABKS

In the SCKA security game, the following scenario will happen [16]. This definition is formalized according to a security game between any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and challenger $\mathcal{C}$.

Setup: Adversary $\mathcal{A}$ selects a challenge I^*_{Enc} and sends it to challenger $\mathcal{C}$. Then challenger $\mathcal{C}$ runs $(MSK, PP) \leftarrow \mathsf{Setup}(\lambda)$ to obtain public parameter PP and master secret key MSK.

Phase 1: Adversary $\mathcal{A}$ is allowed to query the following oracles, and $\mathcal{C}$ keeps an initially empty keyword list L_{KW}. These oracles are as follows:

- $SK \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(I_{\mathsf{KeyGen}})$: If $T(I_{\mathsf{KeyGen}}, I^*_{\mathsf{Enc}}) = 1$, then stop; otherwise, challenger $\mathcal{C}$ returns secret key SK corresponding to I_{KeyGen} to adversary $\mathcal{A}$.
- $st_\omega \leftarrow \mathcal{O}_{\mathsf{TokenGen}}(I_{\mathsf{KeyGen}}, \omega)$: Challenger $\mathcal{C}$ generates secret key SK corresponding to I_{KeyGen} and runs algorithm TokenGen on inputs SK and ω to obtain search token st_ω, and sends this token to adversary $\mathcal{A}$. If $T(I_{\mathsf{KeyGen}}, I^*_{\mathsf{Enc}}) = 1$, then $\mathcal{C}$ adds ω to L_{KW}.

Challenge: Adversary $\mathcal{A}$, selects two keywords ω_0 and ω_1 where $\omega_0, \omega_1 \notin L_{KW}$. Then challenger $\mathcal{C}$ flips a random coin $b \leftarrow \{0,1\}$, and computes challenge ciphertext $(D^*_{\mathsf{cph}}, \mathsf{Index}^*) \leftarrow \mathsf{BuildIndex}(I^*_{\mathsf{Enc}}, \omega_b)$ and sends it to adversary $\mathcal{A}$. If $\omega_0, \omega_1 \in L_{KW}$, adversary $\mathcal{A}$ is prevented to issue his queries to oracle $\mathcal{O}_{\mathsf{TokenGen}}$.

Phase 2: Adversary $\mathcal{A}$ goes on to issue his queries to oracles $\mathcal{O}_{\mathsf{KeyGen}}$ and $\mathcal{O}_{\mathsf{TokenGen}}$, as in the first phase. Note that $(I_{\mathsf{KeyGen}}, \omega_0)$ and $(I_{\mathsf{KeyGEn}}, \omega_1)$ can-not be quired to oracle $\mathcal{O}_{\mathsf{TokenGen}}$ if $T(I_{\mathsf{KeyGen}}, I^*_{\mathsf{Enc}}) = 1$.

Guess: Finally, adversary $\mathcal{A}$ try to guess the value of b and outputs bit b', and wins the game if $b' = b$. In this game, the advantage of adversary $\mathcal{A}$ is defined as follows:

$$Adv^{SCKA}_{\mathcal{A},VABKS}(\lambda) = |\Pr[b' = b] - \frac{1}{2}| \tag{1}$$

Definition 1 (SCKA Security). A VABKS system is selectively secure against chosen keyword attack if the advantage of any PPT adversary $\mathcal{A}$ winning the SCKA game is a negligible function in terms of security parameter λ and

$$Adv^{SCKA}_{\mathcal{A},VABKS}(\lambda) \leq \mathsf{negl}(\lambda) \tag{2}$$

3.1.2 Verifiability Game of VABKS

In the verifiability security game, the following scenario will happen [16]. This definition is formalized according to a security game between PPT adversary $\mathcal{A}$ and challenger $\mathcal{C}$ and both of them are given security parameter λ.

Setup: Challenger $\mathcal{C}$ runs $(PP, MSK) \leftarrow \mathsf{Setup}(\lambda)$ on input security parameter λ. Then adversary $\mathcal{A}$ chooses keyword group KG and access tree I_{Enc} and delivers them to challenger $\mathcal{C}$. Challenger $\mathcal{C}$ runs $(D_{\mathsf{cph}}, \mathsf{Index}) \leftarrow \mathsf{BuildIndex}(I_{\mathsf{Enc}}, KG)$ and sends $(D_{\mathsf{cph}}, \mathsf{Index})$ to $\mathcal{A}$.

Phase 1: Adversary $\mathcal{A}$ is allowed to query the following oracles polynomially many times.

- $SK \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(I_{\mathsf{KeyGen}})$: Challenger $\mathcal{C}$ sends to adversary $\mathcal{A}$ secret key SK associating to I_{KeyGen}. Then this oracle adds I_{KeyGen} to initially empty list L_{KeyGen}.
- $st_\omega \leftarrow \mathcal{O}_{\mathsf{TokenGen}}(I_{\mathsf{KeyGen}}, \omega)$: Challenger $\mathcal{C}$ generates secret key SK associated to I_{KeyGen} and runs algorithm TokenGen where the inputs are secret key SK and queried keyword ω, to generate search token st_ω. Then adversary $\mathcal{C}$ sends it to adversary $\mathcal{A}$.
- $b \leftarrow \mathcal{O}_{\mathsf{Verify}}(I_{\mathsf{KeyGen}}, \omega, st_\omega, \mathsf{rslt}, \mathsf{proof})$: Challenger $\mathcal{C}$ generates secret key SK corresponding to I_{KeyGen}, then runs $b = \mathsf{Verify}(SK, \omega, st_\omega, \mathsf{rslt}, \mathsf{proof})$ and returns b to adversary $\mathcal{A}$.

Challenge: Adversary $\mathcal{A}$ selects I^*_{KeyGen}, satisfying the access control policy which is determined by the data owner and keyword ω^* and gives them to $\mathcal{C}$. Challenger $\mathcal{C}$ randomly chooses I^*_{KeyGen} which satisfies the access tree, i.e. $T(I^*_{KeyGen}, I_{Enc}) = 1$, and generates secret key SK^* corresponding to I^*_{KeyGen} and runs algorithm $st_{\omega^*} \leftarrow \mathsf{TokenGen}(SK^*, \omega^*)$ and returns search token st_{ω^*} to $\mathcal{A}$.

Guess: Finally, adversary $\mathcal{A}$ outputs a pair $(\mathsf{rslt}^*, \mathsf{proof}^*)$ and sends it to challenger $\mathcal{C}$. Challenger $\mathcal{C}$ also runs algorithm $(\mathsf{rslt}, \mathsf{proof}) \leftarrow \mathsf{SearchIndex}(D_{\mathsf{cph}}, \mathsf{Index}, st_{\omega^*})$ to know what is the correct search result. Adversary $\mathcal{A}$ wins the verifiability game, i.e. $\mathsf{Exp}^{\mathsf{Vrfy}}_{\mathcal{A},\mathsf{VABKS}}(\lambda) = 1$, when $1 \leftarrow \mathsf{Verify}(SK^*, \omega^*, st_{\omega^*}, \mathsf{rslt}^*, \mathsf{proof}^*)$ and $\mathsf{rslt} \neq \mathsf{rslt}^*$. The advantage of adversary $\mathcal{A}$ to win the game is defined as follows:

$$Adv^{\mathsf{Vrfy}}_{\mathcal{A},\mathsf{VABKS}}(\lambda) \triangleq \Pr[\mathsf{Exp}^{\mathsf{Vrfy}}_{\mathcal{A},\mathsf{VABKS}}(\lambda) = 1] \tag{3}$$

Definition 2 (Verifiability of VABKS). A VABKS system is verifiable if the advantage of any PPT adversary $\mathcal{A}$ to win the verifiability game is negligible in terns of security parameter λ and

$$Adv^{\mathsf{Vrfy}}_{\mathcal{A},\mathsf{VABKS}}(\lambda) \leq \mathsf{negl}(\lambda) \tag{4}$$

3.2 Hierarchical Identity Based Broadcast Encryption (HIBBE)

Horwitz et.al. for the first time proposed the concept of hierarchical identity-based encryption (HIBE) [21]. In HIBE, public key generator (PKG) shares its burden of generating users' secret keys with the users higher in the hierarchy. Therefore, in the hierarchical model of a network, the users higher in the hierarchy are able to generate secret keys for all of their subordinates. Gentry et.al. [22] presented a HIBE scheme based on the assumption of the difficulty of the bilinear Diffie-Hellman problem. Later, in 2004, Boneh and Boyen [23] presented a HIBE scheme with selective-ID security without needing to the random oracles model. In 2005, Boneh et.al. [24] proposed a secure HIBE with fixed size ciphertext and Gentry et.al. in 2009 [25] constructed the first HIBE for the networks with polynomially numerous hierarchical levels. In 2014, Liu presented a HIBBE to apply existing HIBE systems in multicast and broadcast networks [19] and in 2015, he proposed a practical chosen ciphertext secure HIBBE [26].

Typically, each HIBBE scheme contains four polynomial time algorithms: Setup, Extract, BroadEnc, and BroadDec. These four algorithms are presented in the following.

- $(PP, MSK) \leftarrow \mathsf{Setup}(\lambda)$: This algorithm takes security parameter λ as input and generates master secret key MSK and public parameter PP.
- $USK_{\mathbf{ID}} \leftarrow \mathsf{Extract}(USK_{Par(\mathbf{ID})}, \mathbf{ID}, PP)$: The inputs of this algorithm are $USK_{Par(\mathbf{ID})}$ as the parent's secret key corresponding to identity vector **ID**, identity vector **ID** and public parameter PP. It outputs secret key $USK_{\mathbf{ID}}$ of the user associated to identity vector **ID**. Note that we can consider trusted authority as a user of 0-level hierarchy with identity vector $\mathbf{ID}_0 = ID_{TA}$. So, his secret key is denoted as $USK_0 = MSK$.
- $C \leftarrow \mathsf{BroadEnc}(PP, M, \mathbf{V})$: This algorithm takes public parameter PP, message $M \in \mathcal{M}$, and identity vector set $\mathbf{V}$ as input and encrypt message M where the output is ciphertext C.
- $M := \mathsf{BroadDec}(PP, C, SK_{\mathbf{ID}})$: This algorithm takes as input public parameter PP, ciphertext C, and secret key $SK_{\mathbf{ID}}$. If $\mathbf{ID} \in Par(\mathbf{V})$, then the output of this algorithm is the decryption of C, i.e. M.

3.2.1 Anonymous HIBBE

In the following, we define anonymity against the chosen identity vector set and chosen plaintext attacks (Anon-CIVS-CPA) for the HIBBE. In an anonymous HIBBE scheme, it is required that the adversary does not infer any information about the identities of designated receivers by knowing the ciphertext. This means that the identity of receivers should stay anonymous. As an application of anonymous HIBBE, it should be reminded that this primitive is used as a building block in the proposed generic construction of VABKS.

In Anon-CIVS-CPA security model, adversary $\mathcal{A}$ try to learn some information about the identity of receivers by knowing the ciphertext and is allowed to access the extraction oracle adaptively to obtain the user secret key associated with its chosen identity vector **ID**. Adversary $\mathcal{A}$ is not allowed to query secret key of the parents of the challenged receivers of its selection. We formally define Anon-CIVS-CPA security of HIBBE, through a game between challenger $\mathcal{C}$ and PPT adversary $\mathcal{A}$. Note that both of them are given security parameter λ.

Setup: Challenger $\mathcal{C}$ runs the $(MSK, PP) \leftarrow \mathsf{Setup}(\lambda)$ algorithm on input security parameter λ to generate public parameter PP and master secret key MSK and sends the public parameter PP to adversary $\mathcal{A}$.

Phase 1: Adversary $\mathcal{A}$ is allowed to access to the following oracle on his selected identity vector **ID**.

- $SK_{\mathbf{ID}} \leftarrow \mathcal{O}_{\mathsf{Extract}}(\mathbf{ID})$: In this oracle, adversary $\mathcal{A}$ adaptively issues the identity vector **ID** polynomially many times, to challenger $\mathcal{C}$. Challenger $\mathcal{C}$ selects an empty set of identity vector like $L_{\mathbf{ID}}$ and generates secret key $SK_{\mathbf{ID}}$ associated to the identity vector **ID** and adds **ID** to list $L_{\mathbf{ID}}$ and sends it to adversary $\mathcal{A}$.

Challenge: Adversary $\mathcal{A}$ selects two identity vector sets $\mathbf{V}_0$, $\mathbf{V}_1$ that their parents are not in list $L_{\mathbf{ID}}$, i.e., $L_{\mathbf{ID}} \bigcap \{\bigcup_{i=0,1} Par(V_i)\} = \emptyset$, challenge message $M \in \mathcal{M}$, and sends them to challenger $\mathcal{C}$. Then challenger flips a random coin $b \xleftarrow{\$} \{0,1\}$ and encrypts message M by using identity vector set $\mathbf{V}_b$. Then challenger $\mathcal{C}$ sends challenge ciphertext $C_b = \mathsf{BroadEnc}(PP, M, \mathbf{V}_b)$ to adversary $\mathcal{A}$.

Phase 2: Adversary $\mathcal{A}$ continues issuing his queries to oracle $\mathcal{O}_{\mathsf{Extract}}(\mathbf{ID})$ to receive the secret key of identity vector **ID** such that $\mathbf{ID} \notin Par(\mathbf{V}_0) \cup Par(\mathbf{V}_1)$.

Guess: Finally, adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$ as a guess for the value of b, and wins the game, if $b = b'$.

We define the advantage of the PPT adversary $\mathcal{A}$ in attacking the HIBBE system with security parameter λ as follows:

$$Adv_{\mathcal{A},HIBBE}^{Anon-CIVS-CPA}(\lambda) = |\Pr[b' = b] - \frac{1}{2}| \tag{5}$$

Definition 3 (Anon-CIVS-CPA). A HIBBE is Anon-CIVS-CPA secure against PPT adversary $\mathcal{A}$ if its advantage is as follows:

$$Adv_{\mathcal{A},HIBBE}^{Anon-CIVS-CPA}(\lambda) \leq \mathsf{negl}(\lambda) \tag{6}$$

where the notation $\mathsf{negl}(\lambda)$ is a negligible function in terms of security parameter λ.

3.3 Hierarchical Identity-Based MDVS (HIB-MDVS)

In multi designated verifier signature, a signer determines a specific group of users to sign a message for them and only these designated users could verify the signature by means of their secret key. Typically, in these kinds of signatures, there exist three parties which are PKG, original signer and designated verifiers.

Designated verifier signature (DVS) was introduced for the first time by Jakobsson et.al. in 1996 [27]. They also discussed the security notion of strong DVS (SDVS) and after that, in 2004, this security requirement was defined in a formal way [28]. Laguillaumie et.al. [29], presented an anonymous and efficient DVS construction based on the bilinear map. Desmedt [17], introduced the notion of designated multi verifier signature scheme (DMVS), where a set of legitimate users are able to verify the signature. Laguilaumie et.al. [18], constructed a DMVS scheme where the signer selects a fixed set of designated verifiers to sign a message for them. In 2006, for the first time, Chow [30] presented an identity-based strong multi designated verifier signature (IB-SMDVS). In 2007, Laguillaumie et.al. [31] defined

signer's privacy and presented the first MDVS scheme to protect the signers anonymity without encryption. The security of their scheme reduced to the bilinear Diffie-Hellman problem.

In the following, we will introduce the concept of HIB-MDVS, where the users are in a network with a tree-like construction, and the generation of users' secret keys is in a hierarchical setting, such that the users higher in the hierarchy can generate a secret key for the children of his domain. We introduce the notion of HIB-MDVS and define the security model for its unforgeability and will use this primitive as a building block in our generic construction to establish search and access control policies and provide verifiability.

A HIB-MDVS scheme consists of four polynomial-time algorithms: Setup, Extract, MDVSig and MDVSVrfy. These four algorithms are as follows:

- $(PP, MSK) \leftarrow \mathsf{Setp}(\lambda)$: This algorithm takes security parameter λ as input and generates public parameter PP and master secret key MSK.
- $USK_{\mathbf{ID}} \leftarrow \mathsf{Extract}(USK_{Par(\mathbf{ID})}, \mathbf{ID})$: This algorithm takes as input identity vector **ID** and its parent secret key $USK_{Par(\mathbf{ID})}$ to generate user secret key $USK_{\mathbf{ID}}$ associated to the identity vector **ID**. As mentioned before, $MSK = USK_0$ is the secret key of user with 0-level hierarchy (PKG).
- $\sigma \leftarrow \mathsf{MDVSig}(PP, USK_{\mathbf{ID}_s}, M, \mathbf{V})$: In this algorithm, the signer with identity vector $\mathbf{ID}_s$, generates signature σ for message $M \in \mathcal{M}$ and the authorized users could verify this signature and the users whose identity vectors are in identity vector set **V** are authorized users.
- $\{0,1\} := \mathsf{MDVSVrfy}(PP, \mathbf{ID}_s, USK_{\mathbf{ID}}, \sigma)$: It is a deterministic algorithm which takes as input public parameter PP, identity vector of signer $\mathbf{ID}_s$, multi designated verifier signature σ, and user secret key $USK_{\mathbf{ID}}$ and tests whether the signature σ is a valid signature. If $\mathbf{ID} \in Par(\mathbf{V})$, then the user with identity vector **ID** can check the validity of signature σ. This algorithm returns 1 if, (1) $\mathbf{ID} \in Par(\mathbf{V})$ and (2) signature σ is a valid signature; otherwise, returns 0.

3.3.1 Security Model of HIB-MDVS

In the following, we will define security notion existential forgery against adaptively chosen message attack (EF-CMA) for HIB-MDVS according to the following game which is held between challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

Setup: Challenger $\mathcal{C}$ runs algorithm $(MSK, PP) \leftarrow \mathsf{Setup}(\lambda)$ on security parameter λ as input, to generate public parameter PP and master secret key MSK. Challenger $\mathcal{C}$ selects identity vector $\mathbf{ID}_s$ of the signer and generates its corresponding secret key $SK_{\mathbf{ID}_s}$. Then he sends the identity vector of signer, public parameter PP and security parameter λ to adversary $\mathcal{A}$.

Phase 1: Adversary $\mathcal{A}$ is allowed to query adaptively to the following oracles for polynomially many time queries. The challenger keeps messages list L_M which is initially empty.

- $USK_{\mathbf{ID}} \leftarrow \mathcal{O}_{\mathsf{Extract}}(\mathbf{ID})$: This oracle gets the identity vector **ID** and generates secret key $USK_{\mathbf{ID}}$ and sends it to adversary $\mathcal{A}$. Note that if $\mathbf{ID} \in Par(\mathbf{ID}_s)$, this oracle stops to answer to these queries.
- $\sigma_i \leftarrow \mathcal{O}_{\mathsf{Sign}}(M_i, \mathbf{V})$: The adversary issues message M_i to the signing oracle to receive a valid signature of M_i which is signed for set of designated verifiers **V**. Then this oracle runs sign algorithm $\sigma_i \leftarrow \mathsf{MDVSign}(PP, \mathbf{V}, SK_{\mathbf{ID}_s}, M_i)$ to generate signature σ_i and sends it to adversary $\mathcal{A}$. This oracle adds message M_i to list L_M for each query.

Guess: The adversary tries to output a forgery σ_F as a valid signature of message M_F where never queried before ($M_F \notin L_M$). The adversary wins the game if the signature σ_F is a valid HIB-MDVS signature for the message M_F. The EF-CMA security is defined via the experiment $\mathsf{Exp}^{EF-CMA}_{HIB-MDVS,\mathcal{A}}(\lambda)$ and $\mathsf{Exp}^{EF-CMA}_{HIB-MDVS,\mathcal{A}}(\lambda) = 1$ means that adversary $\mathcal{A}$ wins the game. The advantage of adversary $\mathcal{A}$ in this experiment is defined according to Equation (7).

$$Adv^{EF-CMA}_{HIB-MDVS,\mathcal{A}} = \Pr[\mathsf{Exp}^{EF-CMA}_{HIB-MDVS,\mathcal{A}}(\lambda) = 1] \quad (7)$$

Definition 4 (EF-CMA security of HIB-MDVS). The HIB-MDVS system is EF-CMA secure against PPT adversary $\mathcal{A}$ if the advantage of adversary $\mathcal{A}$ is a negligible function like $\mathsf{negl}(\lambda)$ as follows:

$$Adv^{EF-CMA}_{HIB-MDVS,\mathcal{A}} \leq \mathsf{negl}(\lambda) \quad (8)$$

4 Generic Construction of VABKS

In this section, we present our generic construction which is built based on HIBBE, HIB-MDVS, and BF. In this construction, we use HIB-MDVS to establish the access control policy and provide verifiability. We also used HIBBE for generating a searchable ciphertext for a specific group of data users where their attributes satisfy the access control policy.

4.1 Transformation

The proposed generic construction of VABKS is designed based on identity base primitives. Therefore, the first step in this construction is mapping the set of users' attributes onto a unique identity. Then the data owner designates a specific set of data users - who are authorized to generate a valid search token - and generates a verifiable searchable ciphertext for them by using BF, HIB-MDVS and HIBBE. With the inspiration of Abdalla's transformation of IBE to PEKS [4], HIBBE is used for encrypting the search keywords.

To determine the access control policy, the data owner picks a random number and signs it by using HIB-MDVS, and outsources it in the cloud. Then, one of the delegated data users generates a search token which contains two parts, and sends it to the cloud. The first part plays the role of verifying secret key of outsourced signature. To find the data which are stored for the data user, the cloud receives the search token and starts to verify all of the outsourced signatures in the cloud by using the first part of the search token. If there exists some valid signatures, it means that the documents corresponding to these signatures are stored for the data user and this data user satisfies the access control policy which is determined by data owner.

The data owner, also generates a BF for each keyword group and encrypts it with HIBBE to make the construction verifiable and signs all of the ciphertexts according to the HIB-MDVS to protect the integrity of the messages for the receiver. The cloud runs the search operation on stored encrypted data by means of the received token and sends the search result with the proof of

search accuracy to the data user. The data user first checks the integrity of the received messages and then verifies the search result by means of its secret key and the received proof.

In this transformation, three injective functions are used: $H_0 : \{0,1\}^* \to \mathcal{ID}$, $H_1 : I_{\mathsf{KeyGen}} \to \boldsymbol{\mathcal{ID}}$, and $H_2 : I_{\mathsf{Enc}} \to \boldsymbol{\mathcal{V}}$, where function H_0 maps each string with arbitrary length to an element in identity space $\mathcal{ID}$, function H_1 maps each set of attributes I_{KeyGen} to an element in identity vector space $\boldsymbol{\mathcal{ID}}$ and function H_2 maps each access control policy I_{Enc} to an element in identity vectors space $\boldsymbol{\mathcal{V}}$. In the following, we will explain this construction in detail and express how we can design the six algorithms of VABKS by using the building blocks of HIBBE, HIB-MDVS and BF.

As aforementioned, each VABKS system, contains six polynomially time algorithms which are: $\mathsf{Setup}_{\mathsf{VABKS}}$, KeyGen, BuildIndex, TokenGen, SearchIndex, and Verify. Note that in this transformation, it is assumed that the trusted authority (TA) with zero level of hierarchy knows master secret key MSK.

To compute master secret key MSK and public parameter PP, the TA runs $(MSK, PP) \leftarrow \mathsf{Setup}(\lambda)$ as follows:

$$\begin{aligned} (\mathsf{MDVS.msk}, \mathsf{MDVS.pp}) &\leftarrow \mathsf{Setup}_{\mathsf{HIB-MDVS}}(\lambda) \\ (\mathsf{BE.msk}, \mathsf{BE.pp}) &\leftarrow \mathsf{Setup}_{\mathsf{HIBBE}}(\lambda) \\ MSK &:= (\mathsf{MDVS.msk}, \mathsf{BE.msk}) \quad (9) \\ PP &:= (\mathsf{MDVS.pp}, \mathsf{BE.pp}) \quad (10) \end{aligned}$$

The trusted authority runs key generation algorithm $USK \leftarrow \mathsf{KeyGen}(I_{\mathsf{KeyGen}}, MSK)$, to generate the secret key of the user associated to the attributes set I_{KeyGen}. In the first step, it maps the set of attributes I_{KeyGen} to a unique identity vector $\mathbf{ID}_{\mathsf{KeyGen}}$ by using function H_1 and then acts as follows:

$$\begin{aligned} \mathsf{MDVS.sk} &\leftarrow \mathsf{Extract}_{\mathsf{HIB-MDVS}}(\mathsf{MDVS.msk}, \mathbf{ID}_{\mathsf{KeyGen}}) \\ \mathsf{BE.sk} &\leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.msk}, \mathbf{ID}_{\mathsf{KeyGen}}) \\ USK_{\mathbf{ID}_{\mathsf{KeyGen}}} &:= (\mathsf{MDVS.sk}, \mathsf{BE.sk}) \quad (11) \end{aligned}$$

The data owner runs search index generation algorithm $(\mathsf{Index}, D_{\mathsf{cph}}) \leftarrow \mathsf{BuildIndex}(PP, KG, I_{\mathsf{Enc}})$, according to Algorithm 1. It first computes identity vector set $\mathbf{V}_0 = H_2(I_{\mathsf{Enc}})$ as the set of authorized data users. Then vector set $\mathbf{V}_0$ is also used for encrypting the BF and signing the encrypted ciphertexts to provide their integrity. Then the data owner determines identity vector sets $\mathbf{V}$ and $\mathbf{V}_i$ according to the $\mathbf{V}_0$. $\mathbf{V}$ is used to sign a random message to determine the access control policy. Each data user whose identity vector is in $\mathbf{V}_0$ is able to generate a valid secret key for the cloud to verify the signature, and after verifying, the cloud realizes weather this user satisfies the search control policy. $\mathbf{V}_i$ is determined for each keyword $\omega_i \in KG$, as one of the inputs of algorithm BroadEnc, to generate a searchable ciphertext related to keyword ω.
Finally, the data owner outsources ciphertext vector CPH to the cloud.

For constructing algorithm $st_{\omega^*,j} \leftarrow \mathsf{TokenGen}(SK_j, \omega^*)$, the data user with attributes set I_{KeyGen_j} and secret key SK_j, generates search token $st_{\omega^*,j}$, as depicted in Algorithm 2. As the data user with identity vector $\mathbf{ID}_j$ is the parent of identity vectors $\mathbf{ID}_{j,j}$ and $\mathbf{ID}_{j,\omega^*}$, it is able to generate secret keys associating to these identity vectors. Generated vector key $st_{\omega^*,j}$ is the search token related to keyword ω^* and the data user sends it to the cloud to run the search operation.

The cloud receives search token $st_{\omega^*,j}$ and runs search algorithm $\{(\mathsf{rslt}, \mathsf{proof})\} \leftarrow \mathsf{SearchIndex}(st_{\omega^*,j}, \mathsf{CPH})$ according to Algorithm 3. To check whether the data user with identity vector $\mathbf{ID}_j$ is authorized, it verifies signature $\sigma_{KG} \in \mathsf{CPH}$ by using received key $d_2 \in st_{\omega^*,j}$.

According to Algorithm 3, if $v = 1$, then the user is authorized and the cloud searches for queried keyword on behalf of data user $\mathbf{ID}_j$. The cloud computes result rslt and search proof proof according to two scenarios: (1) The result of search is zero, ($\omega^* \notin KG$). In this case, the result of search is $\mathsf{rslt} = (C_{\mathsf{BF}_{KG}}, \mathsf{BF}'_{KG})$, and the proof is $\mathsf{proof} = (\sigma_{C_{\mathsf{BF}_{KG}}}, \sigma_{\mathsf{BF}'_{KG}})$. (2) Keyword $\omega^* \in KG$ in this case the search result is $\mathsf{rslt} = (C_{\omega^*}, R)$, and the proof is $\mathsf{proof} = (\sigma_{C_{\omega^*}} \sigma_{KG})$. To run the search operation, the cloud decrypts all of the keyword ciphertexts $C_{\omega_i} \in D_{\mathsf{cph}}$. If there exists a keyword ciphertext whit equal decryption to $R \in \mathsf{CPH}$, then the keyword associated to search token $st_{\omega^*,j}$ exists in keyword group KG. To this end, the cloud acts according to the procedure presented in Algorithm 3.

As the last step, the data user with identity vector $\mathbf{ID}_j$ checks the validity of the search results by running algorithm $\{0,1\} \leftarrow \mathsf{Verify}(SK_j, \omega^*, st_{\omega^*,j}, \mathsf{rslt}, \mathsf{proof})$ by means of its secret key SK_j, keyword ω^*, and pair rslt, proof according to Algorithm

Algorithm 1 Search index generation algorithm: $(\mathsf{Index}, D_{\mathsf{cph}}) \leftarrow \mathsf{BuildIndex}(PP, KG, I_{\mathsf{Enc}})$

Input: $(PP, KG, I_{\mathsf{Enc}})$
Output: $(\mathsf{Index}, D_{\mathsf{cph}})$

$\mathbf{V}_0 = H_2(I_{\mathsf{Enc}}) = \{\mathbf{ID}_1, \mathbf{ID}_2, \ldots, \mathbf{ID}_t\}$
$\mathbf{V} = \left\{(\mathbf{ID}_1, H_0(\mathbf{ID}_1)), \ldots, (\mathbf{ID}_t, H_0(\mathbf{ID}_t))\right\}$
$R \xleftarrow{\$} \{0,1\}^\lambda, M \xleftarrow{\$} \{0,1\}^m$
$\sigma_{KG} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}, R, \mathsf{MDVS.sk}_s)$
for $1 \le i \le |KG|$ **do**
  $ID_{\omega_i} = H_0(\omega_i)$
  $\mathbf{V}_i = \left\{(\mathbf{ID}_1, ID_{\omega_i}), \ldots, (\mathbf{ID}_t, ID_{\omega_i})\right\}$
  $C_{\omega_i} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_i, R)$
  $\sigma_{C_{\omega_i}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, C_{\omega_i}, \mathsf{MDVS.sk}_s)$
end for
$\mathsf{BF}_{KG} \leftarrow \mathsf{BFGen}(\{H'_1, \ldots, H'_k\}, KG)$
$C_{\mathsf{BF}_{KG}} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_0, M)$
$\sigma_{C_{\mathsf{BF}_{KG}}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, C_{\mathsf{BF}_{KG}}, \mathsf{MDVS.sk}_s)$
$\mathsf{BF}'_{KG} = M \oplus \mathsf{BF}_{KG}$
$\sigma_{\mathsf{BF}'_{KG}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, \mathsf{BF}'_{KG}, \mathsf{MDVS.sk}_s)$
$D_{\mathsf{cph}} := (C_{\omega_1}, \ldots, C_{\omega_n}, C_{\mathsf{BF}_{KG}}, \mathsf{BF}'_{KG})$
$\mathsf{Index} := (R, \sigma_{KG}, \sigma_{C_{\omega_1}}, \ldots, \sigma_{C_{\omega_n}}, \sigma_{C_{\mathsf{BF}_{KG}}}, \sigma_{\mathsf{BF}'_{KG}})$
return $\mathsf{CPH} := (D_{\mathsf{cph}}, \mathsf{Index})$

Algorithm 2 Token generation algorithm: $st_{\omega^*,j} \leftarrow \mathsf{TokenGen}(SK_j, \omega^*)$

Input: (SK_j, ω^*)
Output: $st_{\omega^*,j}$

$ID_{\omega^*} = H_0(\omega^*)$
$\mathbf{ID}_j = H_1(I_{\mathsf{KeyGen}_j})$
$\mathbf{ID}_{j,\omega^*} = (\mathbf{ID}_j, ID_{\omega^*})$
$\mathbf{ID}_{j,j} = (\mathbf{ID}_j, H_0(\mathbf{ID}_j))$
$d_1 \leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.sk}_j, \mathbf{ID}_{j,\omega^*})$
$d_2 \leftarrow \mathsf{Extract}_{\mathsf{HIB-MDVS}}(\mathsf{MDVS.sk}_j, \mathbf{ID}_{j,j})$
return $st_{\omega^*,j} := (d_1, d_2)$

Algorithm 3 Search algorithm which is done by the cloud: $\{\mathsf{rslt}, \mathsf{proof}\} \leftarrow \mathsf{SearchIndex}(st_{\omega^*,j}, \mathsf{CPH})$

Input: $(st_{\omega^*,j}, \mathsf{CPH})$
Output: $(\{\mathsf{rslt}, \mathsf{proof}\})$

$v = \mathsf{MDVSVrfy}(\mathsf{MDVS.pp}, \mathbf{ID}_{owner}, \sigma_{KG}, d_2)$
if $v == 1$ **then**
  $F(I_{\mathsf{KeyGen}}, I_{\mathsf{Enc}}) = 1$
  for $1 \leq i \leq |KG|$ **do**
    $R_i = \mathsf{BroadDec}(\mathsf{BE.pp}, d_1, C_{\omega_i})$
    $\mathcal{R} = \mathcal{R} \cup R_i$
  end for
  if $\exists R_i \in \mathcal{R}$ s.t. $R_i == R$ **then**
    $\omega^* \in KG$&
    $\mathsf{rslt} = (C_{\omega_i}, R)$&
    $\mathsf{proof} = (\sigma_{C_{\omega_i}}, \sigma_{KG})$
  else
    $\omega^* \notin KG$
    $\mathsf{rslt} = (C_{\mathsf{BF}_{KG}}, \mathsf{BF}'_{KG})$
    $\mathsf{proof} = (\sigma_{C_{\mathsf{BF}_{KG}}}, \sigma_{\mathsf{BF}'_{KG}})$
  end if
else
  $F(I_{\mathsf{KeyGen}}, I_{\mathsf{Enc}}) = 0$
  $\mathsf{rslt} = \perp$ & $\mathsf{proof} = \perp$
end if
return $\{(\mathsf{rslt}, \mathsf{proof})\}$

Algorithm 4 The verification algorithm which is run by data user: $\{0,1\} := \mathsf{Verify}(SK_j, \omega^*, st_{\omega^*,j}, \mathsf{rslt}, \mathsf{proof})$

Input: $(SK_j, \omega^*, st_{\omega^*,j}, \mathsf{rslt}, \mathsf{proof})$
Output: $\{0,1\}$
{The data user checks the integrity of the received messages}

if $\omega^* \in KG$ **then**
  $R' = \mathsf{BoradDec}(\mathsf{BE.pp}, d_1, C_{\omega^*})$
  if $R' == R$ **then**
    $v' = 1$
  else
    $v' = 0$
  end if
else
  $M = \mathsf{BroadDec}(\mathsf{BE.pp}, C_{\mathsf{BF}_{KG}}, \mathsf{BE.sk}_j)$
  $\mathsf{BF}_{KG} = \mathsf{BF}'_{KG} \oplus M$ {Where $\mathsf{BF}'_{KG} \in \mathsf{rslt}$ }
  $v_{\mathsf{BF}} \leftarrow \mathsf{BFverify}(\mathsf{BF}_{KG}, \omega^*)$
  if $v_{\mathsf{BF}} == 1$ **then**
    $v' = 0 \Rightarrow \omega^* \notin KG$
  else
    $v' = 1$
  end if
end if
return v'

4. If $\omega^* \in KG$, the data user, after verifying the integrity of received pair $\{(\mathsf{rslt}, \mathsf{proof})\}$, runs the search operation on result rslt. If keyword $\omega^* \notin KG$, then the result contains the ciphertext of BF_{KG} and the proof contains the signature of the ciphertext of BF_{KG}. In this case, the data user first verifies the integrity of rslt, by using proof proof, then decrypts the ciphertext of BF_{KG}, then it runs algorithm BFverify to check the existence of keyword ω^* in the keyword group associated to this BF. This procedure is presented in Algorithm 4 with more details.

In Algorithm 4, if $v' = 1$, it means that the cloud runs the search operation honestly and the received search result are valid.

In the next section, we analyze the security of our construction.

5 Security Proof

In this section, we will prove that the proposed generic construction of VABKS is verifiable and secure against selectively chosen keyword attack. In what follows, we will prove if there exists a HIBBE with Anon-HIBBE-CPA security, then the resulting VABKS is SCKA-secure against all PPT adversaries and if there exists a HIBBE with Anon-HIBBE-CPA security and HIB-MDVS with EF-CMA security, then the constructed VABKS is secure in the verifiability game.

Theorem 1 (SCKA-Security). If there exists a HIBBE system with Anon-CIVS-CPA security, then the advantage of PPT adversary $\mathcal{A}$ to win the SCKA game is a negligible function and we have:

$$Adv^{SCKA}_{\mathcal{A},VABKS} \leq \mathsf{negl}(\lambda) \tag{12}$$

Proof. We will show that if the advantage PPT adversary $\mathcal{A}$ to win the SCKA game be a non-negligible function, then there exists a PPT adversary $\mathcal{B}$ who wins the Anon-CIVS-CPA security game with a non-negligible advantage. So, adversary $\mathcal{B}$ simulates the SCKA game to win the Anon-CIVS-CPA security game as follows.

Setup: Adversary $\mathcal{A}$ selects challenge access control policy I^*_{Enc} and gives it to adversary $\mathcal{B}$ ($\mathcal{B}$ plays the role of the challenger in the view of adversary $\mathcal{A}$). Challenger $\mathcal{B}$ computes $\mathbf{V}^* = H_2(I^*_{\mathsf{Enc}}) = \{\mathbf{ID}^*_1, \ldots, \mathbf{ID}^*_t\}$. According to security parameter λ, challenger of Anon-CIVS-CPA, runs algorithm $(\mathsf{BE.pp}, \mathsf{BE.msk}) \leftarrow \mathsf{Setup}_{\mathsf{HIBBE}}(\lambda)$ and sends public parameter BE.pp to adversary $\mathcal{B}$ and protects master secret key BE.msk. Then adversary $\mathcal{B}$ runs algorithm $(\mathsf{MDVS.pp}, \mathsf{MDVS.msk}) \leftarrow \mathsf{Setup}_{\mathsf{HIB-MDVS}}(\lambda)$ and sends public parameters BE.pp and MDVS.pp to adversary $\mathcal{A}$.

Phase 1: Adversary $\mathcal{A}$ can query the following oracles adaptively and polynomially many times. Note that adversary $\mathcal{B}$ keeps keyword list L_{kw} which is initially empty.

- $\mathcal{O}_{\mathsf{KeyGen}}(I_{\mathsf{KeyGen}})$: Adversary $\mathcal{B}$ receives attributes set I_{KeyGen} from adversary $\mathcal{A}$ and computes $\mathbf{ID} = H_1(I_{\mathsf{KeyGen}})$. If $\mathbf{ID} \in Par(\mathbf{V}^*)$ then adversary $\mathcal{B}$ stops; otherwise, $\mathcal{B}$ sends identity vector $\mathbf{ID}$ to challenger and receives secret key $\mathsf{BE.sk}_{\mathbf{ID}} \leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.msk}, \mathbf{ID})$ corresponding to attribute set I_{KeyGen}. Then adversary $\mathcal{B}$ runs algorithm $\mathsf{MDVS.sk}_{\mathbf{ID}} \leftarrow \mathsf{Extract}_{\mathsf{HIB-MDVS}}(\mathsf{MDVS.msk}, \mathbf{ID})$. The resulting secret key associated to $\mathbf{ID}$ is $SK_{\mathbf{ID}} := (\mathsf{MDVS.sk}_{\mathbf{ID}}, \mathsf{BE.sk}_{\mathbf{ID}})$ and is sent to adversary $\mathcal{A}$.
- $\mathcal{O}_{\mathsf{TokenGen}}(I_{\mathsf{KeyGen}}, \omega)$: Adversary $\mathcal{B}$ receives attribute set I_{KeyGen}, and keyword ω form adversary $\mathcal{A}$ and it should compute search token st_ω for $\mathcal{A}$. So, adversary $\mathcal{B}$ computes $\mathbf{ID} = H_1(I_{\mathsf{KeyGen}})$ and sends identity vector $\mathbf{ID}$ to challenger $\mathcal{C}$ and he sets identity vector $\mathbf{ID}_\omega = (\mathbf{ID}, H_0(\omega))$. The challenger computes secret key $\mathsf{BE.sk}_{\mathbf{ID}} \leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.msk}, \mathbf{ID})$ and sends this

TABLE 2
Required building blocks in proposed generic construction of VABKS. For example, to construct algorithm Setup of VABKS two algorithms $\mathsf{Setup}_{\mathsf{HIBBE}}$ and $\mathsf{Setup}_{\mathsf{HIB-MDVS}}$ are required.

	The building blocks of VABKS					
	Setup	KeyGen	BuildIndex	TokenGen	SearchIndex	Verify
$\mathsf{Setup}_{\mathsf{HIBBE}}$	✓	-	-	-	-	-
$\mathsf{Setup}_{\mathsf{HIB-MDVS}}$	✓	-	-	-	-	-
$\mathsf{Extract}_{\mathsf{HIBBE}}$	-	✓	-	✓	-	-
$\mathsf{Extract}_{\mathsf{HIB-MDVS}}$	-	✓	-	✓	-	-
BroadEnc	-	-	✓	-	-	-
BroadDec	-	-	-	-	✓	✓
MDVSig	-	-	✓	-	-	-
MDVSVrfy	-	-	-	-	✓	✓
BFGen	-	-	✓	-	-	-
BFVerify	-	-	-	-	-	✓
$\mathbf{H}_0$	-	-	✓	✓	-	✓
$\mathbf{H}_1$	-	✓	-	-	-	-
$\mathbf{H}_2$	-	-	✓	-	-	-

secret key to adversary $\mathcal{B}$. Adversary $\mathcal{B}$ runs algorithm $\mathsf{MDVS.sk}_{\mathbf{ID}} \leftarrow \mathsf{Extract}_{\mathsf{HIB-MDVS}}(\mathsf{MDVS.msk}, \mathbf{ID})$, computes search token $st_\omega \leftarrow \mathsf{Extract}(SK_{ID}, \mathbf{ID}_\omega)$ where $SK_{\mathbf{ID}} = (\mathsf{MDVS.sk}_{\mathbf{ID}}, \mathsf{BE.sk}_{\mathbf{ID}})$ and sends st_ω to adversary $\mathcal{A}$. If $\mathbf{ID} \in Par(\mathbf{V}^*)$ then the adversary adds keyword ω to keyword list L_{kw}.

Challenge Phase: Adversary $\mathcal{A}$ chooses two keywords ω_0 and ω_1 ($\omega_0, \omega_1 \notin L_{kw}$) and sends them to adversary $\mathcal{B}$. In this case, adversary $\mathcal{B}$ computes $H_0(\omega_0)$ and $H_0(\omega_1)$ and sets two identity vectors $\mathbf{V}_0 = \{(\mathbf{ID}_1^*, H_0(\omega_0)), \ldots, (\mathbf{ID}_t^*, H_0(\omega_0))\}$ and $\mathbf{V}_1 = \{(\mathbf{ID}_1^*, H_0(\omega_1)), \ldots, (\mathbf{ID}_t^*, H_0(\omega_1))\}$. Then it selects string $R \xleftarrow{\$} \{0,1\}^\lambda$ uniformly at random, as the message that challenger $\mathcal{C}$ wants to encrypt. Then adversary $\mathcal{B}$ sends R, $\mathbf{V}_0$ and $\mathbf{V}_1$ to the challenger. The challenger selects a random bit $b \xleftarrow{\$} \{0,1\}$ and computes $C_b \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_b, R)$ and sends it to adversary $\mathcal{B}$. Adversary $\mathcal{B}$ sets $C^* := C_b$ and invokes adversary $\mathcal{A}$ to guess the correct value of bit b where C^* is the challenge keyword ciphertext in SCKA game.

Phase 2: Adversary $\mathcal{A}$ continues to query the same as the first phase, but he could not query $(I_{\mathsf{KeyGen}}, \omega_0)$ and $(I_{\mathsf{KeyGen}}, \omega_1)$ to oracle $\mathcal{O}_{\mathsf{TokenGen}}(.,.)$ if the $H_1(I_{\mathsf{KeyGen}}) \in Par(\mathbf{V}^*)$.

Guess: In this part, adversary $\mathcal{B}$ should guess the value of b. To do this, after adversary $\mathcal{A}$ outputs the value of b' he sends it to challenger $\mathcal{C}$ as a guess for b. We supposed that adversary $\mathcal{A}$ wins the SCKA game with non-negligible advantage, therefore we can conclude that adversary $\mathcal{B}$ wins the Anon-CIVS-CPA security game with non-negligible advantage.

In what follows, we will compute the advantage of adversary $\mathcal{B}$ in the Anon-CIVS-CPA game.

$$Adv^{SCKA}_{\mathcal{A},VABKS} \triangleq \epsilon(\lambda) \tag{13}$$

$$\begin{aligned} Adv^{Anon-CIVS-CPA}_{\mathcal{B},HIBBE} &= |\Pr[b' = b] - \frac{1}{2}| \\ &= Adv^{SCKA}_{\mathcal{A},VABKS} = \epsilon(\lambda) \\ &\Rightarrow Adv^{Anon-CIVS-CPA}_{\mathcal{B},HIBBE} = \epsilon(\lambda) \end{aligned}$$

In Equation (13), function $\epsilon(\lambda)$ is a non-negligible function. According to the above equations, it is obvious that the advantage of adversary $\mathcal{B}$ to win the Anon-CIVS-CPA game is non-negligible function in terms of security parameter λ where this contradicts with the assumption that the HIBBE system is Anon-CIVS-CPA secure. So, the constructed VABKS is SCKA secure.

Theorem 2 (Verifiability). If there exists a HIB-MDVS system with EF-CMA security, then the resulting VABKS from the proposed generic construction is verifiable against PPT adversary $\mathcal{A}$.

Proof. In this proof, it is supposed that the constructed VABKS is not verifiable. So, there exists a PPT adversary like $\mathcal{A}$ who can find a pair $\{\mathsf{rslt}, \mathsf{proof}\}$ where rslt is not a correct result and the output of algorithm Verify is 1, with non-negligible advantage in terms of security parameter λ. Then it is shown that there is a PPT adversary like $\mathcal{B}$, who wins the EF-CMA experiment with non-negligible probability. Adversary $\mathcal{B}$ invokes adversary $\mathcal{A}$ to win EF-CMA experiment.

Setup: Challenger $\mathcal{C}$ runs $(\mathsf{MDVS.msk}, \mathsf{MDSVS.pp}) \leftarrow \mathsf{Setup}_{\mathsf{HIB-MDVS}}(\lambda)$ and sends security parameter λ, public parameter MDSVS.pp, and identity vector of the data owner $\mathbf{ID}_s$ to adversary $\mathcal{B}$. The challenger generates secret key $\mathsf{MDVS.sk}_s$ of the data owner. Adversary $\mathcal{B}$ runs $(\mathsf{BE.msk}, \mathsf{BE.pp}) \leftarrow \mathsf{Setup}_{\mathsf{HIBBE}}(\lambda)$ and sends $PP = (\mathsf{MDVS.pp}, \mathsf{BE.pp})$ and security parameter λ to $\mathcal{A}$. Adversary $\mathcal{A}$ selects I_{Enc} and keyword group KG and sends them to adversary $\mathcal{B}$. Then adversary $\mathcal{B}$ determines identity vector set $\mathbf{V}_0 = \{\mathbf{ID}_1, \ldots, \mathbf{ID}_t\} = H_2(I_{\mathsf{Enc}})$ and generates verifiable searchable ciphertext $(D_{\mathsf{cph}}, \mathsf{Index}) \leftarrow \mathsf{BuildIndex}(KG, I_{\mathsf{Enc}})$ and sends it to adversary $\mathcal{A}$. To generate the ciphertexts and index, he acts as follows:

$$\begin{aligned} \mathbf{V} &= \left\{(\mathbf{ID}_1, H_0(\mathbf{ID}_1)), \ldots, (\mathbf{ID}_t, H_0(\mathbf{ID}_t))\right\} \\ R &\xleftarrow{\$} \{0,1\}^\lambda; M \xleftarrow{\$} \{0,1\}^m \\ \forall \omega_i \in KG &: \{ \\ \mathbf{V}_i &= \left\{(\mathbf{ID}_1, H_0(\omega_i)), \ldots, (\mathbf{ID}_t, H_0(\omega_i))\right\} \\ C_{\omega_i} &\leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_i, R) \\ \} & \\ C_{BF_{KG}} &\leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_0, M) \\ \mathsf{BF} &\leftarrow \mathsf{BFGen}(\{H'_1, \ldots, H'_k\}, KG) \\ \mathsf{BF}'_{KG} &:= M \oplus \mathsf{BF} \end{aligned}$$

Then adversary $\mathcal{B}$ queries R and the encrypted messages to the signing oracle according to the VABKS construction to receive the following signatures:

$$\begin{aligned}
\sigma_{KG} &\leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}, R, \mathsf{MDVS.sk}_s)\\
\sigma_{C_{\omega_i}} &\leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_i, C_{\omega_i}, \mathsf{MDVS.sk}_s)\\
\sigma_{C_{BF_{KG}}} &\leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, C_{BF_{KG}}, \mathsf{MDVS.sk}_s)\\
\sigma_{BF'_{KG}} &\leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, \mathsf{BF}'_{KG}, \mathsf{MDVS.sk}_s)
\end{aligned}$$

Note that random value R is signed for determining the access control policy and $1 \leftarrow \mathsf{MDVSVrfy}(\sigma_{KG}, \mathbf{ID}_s, \mathsf{MDVS.sk}_{(\mathbf{ID}_j, H_0(\mathbf{ID}_j))}, R)$ means that the user with identity vector $\mathbf{ID}_j$ satisfies the access control policy. After receiving the signatures from signing oracle, adversary $\mathcal{B}$ sets $(\mathsf{Index}, D_{\mathsf{cph}})$ as follows and sends them to adversary $\mathcal{A}$.

$$\begin{aligned}
D_{\mathsf{cph}} &= (C_{\omega_1}, \ldots, C_{\omega_n}, C_{\mathsf{BF}_{KG}}, \mathsf{BF}'_{KG})\\
\mathsf{Index} &= (R, \sigma_{KG}, \sigma_{C_{\omega_1}}, \ldots, \sigma_{C_{\omega_n}}, \sigma_{C_{\mathsf{BF}_{KG}}}, \sigma_{\mathsf{BF}'_{KG}})
\end{aligned}$$

Phase1: In this phase, adversary $\mathcal{B}$ should simulate three oracles $\mathcal{O}_{\mathsf{KeyGen}}, \mathcal{O}_{\mathsf{Extract}}, \mathcal{O}_{\mathsf{Vrfy}}$ for adversary $\mathcal{A}$ by using the signing and extraction oracles of the experiment $\mathsf{Exp}^{EF-CMA}_{\mathcal{B},HIB-MDVS}(\lambda)$. So, adversary $\mathcal{A}$ queries the mentioned oracles as follows:

- $\mathcal{O}_{\mathsf{KeyGen}}(I_{\mathsf{KeyGen}})$: Adversary $\mathcal{A}$ sends I_{KeyGen} to adversary $\mathcal{B}$. Adversary $\mathcal{B}$ computes $\mathbf{ID} = H_1(I_{\mathsf{KeyGen}})$ and sends it to the extraction oracle and receives related secret key $\mathsf{MDVS.sk}_{\mathbf{ID}}$. Then adversary $\mathcal{B}$ runs algorithms $(\mathsf{BE.msk}, \mathsf{BE.pp}) \leftarrow \mathsf{Setup}_{\mathsf{HIBBE}}(\lambda)$ and $\mathsf{BE.sk}_{\mathbf{ID}} \leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.msk}, \mathbf{ID})$ and sends $SK_{\mathbf{ID}} = (\mathsf{MDVS.sk}_{\mathbf{ID}}, \mathsf{BE.sk}_{\mathbf{ID}})$ to adversary $\mathcal{A}$.
- $\mathcal{O}_{\mathsf{TokenGen}}(I_{\mathsf{KeyGen}_j}, \omega)$: Adversary $\mathcal{B}$ receives I_{KeyGen_j} and keyword ω and generates a search token. $\mathcal{B}$ computes $\mathbf{ID}_j = H_1(I_{\mathsf{KeyGen}_j})$ and sends this identity to challenger $\mathcal{C}$ and receives secret key $\mathsf{MDVS.sk}_{\mathbf{ID}_j}$ and generates search token $st_{\omega,\mathbf{ID}_j}$ as follows:

$$\begin{aligned}
\mathbf{ID}_{\omega} &= (\mathbf{ID}_j, H_0(\omega))\\
\mathbf{ID}_{j,j} &= (\mathbf{ID}_j, H_0(\mathbf{ID}_j))\\
\mathsf{BE.sk}_{\mathbf{ID}_j} &\leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.msk}, \mathbf{ID}_j)\\
d_1 &\leftarrow \mathsf{Extract}_{\mathsf{HIBBE}}(\mathsf{BE.sk}_{\mathbf{ID}_j}, \mathbf{ID}_{\omega})\\
d_2 &\leftarrow \mathsf{Extract}_{\mathsf{MDVS}}(\mathsf{MDVS.sk}_{\mathbf{ID}_j}, \mathbf{ID}_{j,j})\\
st_{\omega,\mathbf{ID}_j} &:= (d_1, d_2)
\end{aligned}$$

- $\mathcal{O}_{\mathsf{Vrfy}}(I_{\mathsf{KeyGen}}, \omega, st, \mathsf{rslt}, \mathsf{proof})$: Adversary $\mathcal{B}$ computes $\mathbf{ID} = H_1(I_{\mathsf{KeyGen}})$ and sends it to challenger $\mathcal{C}$ and receives secret key $\mathsf{MDVS.sk}_{\mathbf{ID}}$ and generates secret key $\mathsf{BE.sk}_{\mathbf{ID}}$. Then it runs algorithm $\nu \leftarrow \mathsf{Vrfy}(SK_{\mathbf{ID}}, \omega, st, \mathsf{rslt}, \mathsf{proof})$ and sends ν to adversary $\mathcal{A}$ where $SK_{\mathbf{ID}} = (\mathsf{MDVS.sk}_{\mathbf{ID}}, \mathsf{BE.sk}_{\mathbf{ID}})$.

Challenge Phase: Adversary $\mathcal{A}$ selects challenge access control policy I^*_{Enc} and challenge keyword ω^* and sends them to adversary $\mathcal{B}$. Adversary $\mathcal{B}$ first computes $\mathbf{V}^* = \{\mathbf{ID}^*_1, \ldots, \mathbf{ID}^*_t\} = H_2(I^*_{\mathsf{Enc}})$, then selects identity vector $\mathbf{ID}^*_j \in \mathbf{V}^*$, and then sends it to the extraction oracle to receive secret key $\mathsf{MDVS.sk}_{\mathbf{ID}^*_j}$. Next, it generates $\mathsf{BE.sk}_{\mathbf{ID}^*_j}$ and sets $SK_{\mathbf{ID}^*} = (\mathsf{MDVS.sk}_{\mathbf{ID}^*_j}, \mathsf{BE.sk}_{\mathbf{ID}^*_j})$. After that, adversary $\mathcal{B}$ uses secret key $SK_{\mathbf{ID}^*}$ to generate search token $st_{\omega^*,\mathbf{ID}^*_j} \leftarrow \mathsf{TokenGen}(SK_{\mathbf{ID}^*_j}, \omega^*)$ and sends this token to adversary $\mathcal{A}$.

Guess: Adversary $\mathcal{A}$ outputs result and proof pair $(\mathsf{rslt}^*, \mathsf{proof}^*)$ and sends them to adversary $\mathcal{B}$. It has been supposed that adversary $\mathcal{A}$ wins the game with non-negligible advantage and its advantage of winning the game is $\Pr[1 \leftarrow \mathsf{Vrfy}(SK_{ID^*}, \omega^*, st_{\omega^*,\mathbf{ID}^*_j}, \mathsf{rslt}^*, \mathsf{proof}^*)] = \epsilon(\lambda)$ where function $\epsilon(\lambda)$ is a non-negligible function in terms of security parameter λ. Note that adversary $\mathcal{A}$ wins the game if $\mathsf{rslt} \neq \mathsf{rslt}^*$ where $(\mathsf{rlst}, \mathsf{proof}) \leftarrow \mathsf{SearchIndex}(D_{\mathsf{cph}}, \mathsf{Index}, st_{\omega^*,\mathbf{ID}^*_j})$. There are two states for ω^* when adversary $\mathcal{A}$ wins the game, which are mentioned as follows:

- **State 1:** If $\omega \notin KG$, in this case, rslt is the ciphertext of BF_{KG} and its signature is in proof. So, adversary $\mathcal{A}$ should encrypt the BF as rslt^* and make a valid signature for it as proof^* to win this game. It means that adversary $\mathcal{A}$ can find a forgery of message rslt^* which has never been queried before to the signing oracle. So, adversary $\mathcal{B}$ sends tuple $(\mathsf{rslt}^*, \mathsf{proof}^*, \mathbf{V}^*_0)$ to the challenger. Thus, adversary $\mathcal{B}$ wins experiment $\mathsf{Exp}^{EF-CMA}_{\mathcal{B},HIB-MDVS}(\lambda)$ with a non-negligible function.
- **State 2:** If $\omega \notin KG$, rslt is keyword ciphertext C_{ω^*}. So, rslt^* must be a different keyword ciphertext for ω^*. In this case, the adversary should make a valid signature on the new ciphertext. The same as previous state, it means that adversary $\mathcal{A}$ can break unforgeability of HIB-MDVS.

In what follows, we will compute the advantage of adversary $\mathcal{B}$ to win this experiment.

$$\Pr[\mathsf{Exp}^{\mathsf{Vrfy}}_{\mathcal{A},VABKS}] = \epsilon(\lambda) \tag{14}$$

$$\begin{aligned}
&\Pr[\mathsf{Exp}^{EF-CMA}_{\mathcal{B},HIB-MDVS} = 1]\\
&= \Pr[\mathcal{B}\ \ win]\\
&= \Pr[1 \leftarrow \mathsf{Vrfy}(SK_{ID^*}, \omega^*, st_{\omega^*}, \mathsf{rslt}^*, \mathsf{proof}^*) \wedge \mathsf{rslt} \neq \mathsf{rslt}^*]\\
&= \Pr[\mathsf{Exp}^{\mathsf{Vrfy}}_{\mathcal{A},VABKS}] = \epsilon(\lambda)\\
&\Rightarrow \Pr[\mathsf{Exp}^{EF-CMA}_{\mathcal{B},HIB-MDVS} = 1] = \epsilon(\lambda)
\end{aligned}$$

So, the advantage of adversary $\mathcal{B}$ to find a forgery for the HIB-MDVS scheme is a non-negligible function which contradicts the signature's assumption of EF-CMA security. So, the VABKS scheme, which is constructed from the proposed generic construction, is verifiable.

6 Verifiable Ranked Keyword Search over Outsourced Encrypted Data in Cloud Using VABKS

Searchable encryption schemes are commonly used in cloud computing to provide a secure environment for cloud storage. In some cases, the data users would like to receive the most relevant documents to the queried keywords. For this aim, the suggested solution is secure ranked keyword search over encrypted cloud data which was defined for the first time in [32]. In the ranked keyword search schemes, the score of each file which includes the queried keyword is computed based on the number of times the term appears in this file. The cloud provider searches on encrypted data and returns the data users the top k-ranked documents (The value of k is chosen by data user). These approaches enhance system usability, because they ensure the data users that they have received the most relevant documents to the queried keyword.

In 2014, Cao et.al. introduced the concept of multi-keyword ranked search and proposed a secure construction where the data user is able to send a search query related to a set of keywords to the cloud [33], [34]. In another work, in 2014, Li et.al. presented an efficient multi-keyword ranked search scheme over encrypted mobile cloud data through blind storage [35]. For verifying the accuracy of received search results, in 2014, Sun et.al. proposed a verifiable multi-keyword text search in the cloud which is supporting similarity-based ranking. In their proposed scheme, the keywords of each document are extracted as a vector and the cloud try to find the k-top similar keyword vector to the queried terms [36]. Recently, some other works have been done to apply the semantic search to the multi-keyword ranked search schemes and improve their efficiency and enhance their security, e.g. [37] and [38].

In this section, as the example of one application of proposed generic construction of VABKS it will be shown that this generic scheme can be used to construct a verifiable ranked keyword search scheme over encrypted cloud data. In what follows the construction of keyword ranked search is presented in more details. To adjust the proposed VABKS scheme for this application we will change algorithm BuildIndex to RanBuildIndex according to Algorithm 5. The performance of algorithm RanBuildIndex is the same as BuildIndex and the only different is that in RanBuildIndex, the score of each keyword is also encrypted and forms one part of the encryption of keyword. Typically, each keyword ranked search consists of two phases setup and retrieval as follows.

- **Setup phase:** TA runs algorithm $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{Setup}(\lambda)$ and protects master secret key msk and broadcasts public parameter pp and generates the secret keys of all of the users and distributes the generated keys among them through a secure and authenticated channel. Then the data owner determines the search and access control policy and encrypts the documents according to the access control policy and generates a secure search index according to the search policy and outsources them to the cloud. Note that access and search control policy are not necessarily equal. The detail are given in Table 3. In Table 3, I_{Enc} and $I_{\mathsf{Enc}'}$ respectively denote search and access control policy.
- **Retrieval phase:** The data users U_j runs algorithm $st_{\omega^*,j} \leftarrow \mathsf{TokenGen}(\mathsf{SK}_{U_j}, \omega^*)$ and send search token $st_{\omega^*,j}$ to the cloud. The cloud receives the search token and runs algorithm $\{\mathsf{rslt}, \mathsf{proof}\} \leftarrow \mathsf{SearchIndex}(st_{\omega^*,j}, \mathsf{CPH})$ on the stored encrypted data and returns user U_j the results of search. The data user verifies the correctness of search results by running algorithm $\{0,1\} := \mathsf{Verify}(\mathsf{SK}_j, \omega^*, st_{\omega^*,j}, \mathsf{rslt}, \mathsf{proof})$. If the results of search is valid and the cloud server has founded some documents which are related to the queried keyword, the data user decrypts the second part of C_{ω^*}, i.e., C_{S^*} and choses the top ranked documents and sends the cloud $id(F_{i_1}), \ldots, id(F_{i_k})$ to receive the required documents.

By using verifiable ranked keyword search we can verify the cloud performance and the integrity of stored messages and received the most relevant documents to the queried keyword. We can also reduce the communication cost because the cloud doesn't send the data users all of the documents which contain the queried keyword and just returns back the top-k most relevant documents. The comparison of existing ranked keyword search schemes is presented in Table 4. According to Table 4 all of the mentioned schemes in this table except our proposed scheme need an interaction between the data owner and the data users to share a secret key which is used for generating the search trapdoors. In our scheme, the data users use their secret key which are received form TA to generate the search tokens. As a result of this situation, we can reduce the communication overhead of ranked keyword search schemes.

Algorithm 5 Search index generation algorithm: $(\mathsf{Index}, D_{\mathsf{cph}}) \leftarrow \mathsf{RankBuildIndex}(PP, (KG, S), (I_{\mathsf{Enc}}, I_{\mathsf{Enc}'}))$

Input: $(PP, KG, I_{\mathsf{Enc}})$
Output: $(\mathsf{Index}, D_{\mathsf{cph}})$

$\mathbf{V}_0 = \{\mathbf{ID}_1, \mathbf{ID}_2, \ldots, \mathbf{ID}_t\}$
$\mathbf{V}'_0 = H_2(I_{\mathsf{Enc}'})$ { $I_{\mathsf{Enc}'}$ is the access control policy.}
$\mathbf{V} = \left\{(\mathbf{ID}_1, H_0(\mathbf{ID}_1)), \ldots, (\mathbf{ID}_t, H_0(\mathbf{ID}_t))\right\}$
$R \xleftarrow{\$} \{0,1\}^{\lambda}, M \xleftarrow{\$} \{0,1\}^m$
$\sigma_{KG} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \tilde{V}, R, \mathsf{MDVS.sk}_s)$
for $1 \le i \le |KG|$ **do**
  $ID_{\omega_i} = H_0(\omega_i)$
  $\mathbf{V}_i = \left\{(\mathbf{ID}_1, H_0(\omega_i)), \ldots, (\mathbf{ID}_t, H_0(\omega_i))\right\}$
  $C'_{\omega_i} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_i, R)$
  $C_{S_i} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}'_0, S_i)$ {$S_i \in S$ and S is computed according to Table 3}
  $C_{\omega_i} := C'_{\omega_i} || C_{S_i}$
  $\sigma_{C_{\omega_i}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, C_{\omega_i}, \mathsf{MDVS.sk}_s)$
end for
$\mathsf{BF}_{KG} \leftarrow \mathsf{BFGen}(\{H'_1, \ldots, H'_k\}, KG)$
$C_{\mathsf{BF}_{KG}} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}_0, M)$
$\sigma_{C_{\mathsf{BF}_{KG}}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, C_{\mathsf{BF}_{KG}}, \mathsf{MDVS.sk}_s)$
$\mathsf{BF}'_{KG} = M \oplus \mathsf{BF}_{KG}$
$\sigma_{\mathsf{BF}'_{KG}} \leftarrow \mathsf{MDVSig}(\mathsf{MDVS.pp}, \mathbf{V}_0, \mathsf{BF}'_{KG}, \mathsf{MDVS.sk}_s)$
$D_{\mathsf{cph}} := (C_{\omega_1}, \ldots, C_{\omega_n}, C_{\mathsf{BF}_{KG}}, \mathsf{BF}'_{KG})$
$\mathsf{Index} := (R, \sigma_{KG}, \sigma_{C_{\omega_1}}, \ldots, \sigma_{C_{\omega_n}}, \sigma_{C_{\mathsf{BF}_{KG}}}, \sigma_{\mathsf{BF}'_{KG}})$
return $\mathsf{CPH} := (D_{\mathsf{cph}}, \mathsf{Index})$

7 Conclusion

In this paper, a new cryptographic notion which is called HIB-MDVS was introduced and its security definition was defined based on EF-CMA security game. Furthermore, a formal definition of HIBBE's i.e., anonymity against chosen identity vector set and chosen message attack, was presented. The first generic construction for VABKS in a modular structure was also proposed based on BF, HIBBE, and HIB-MDVS as the required building blocks. Then, it was shown that the security of proposed scheme is based on the unforgeability of HIB-MDVS and the anonymity of HIBBE. We also used our generic construction to construct a verifiable ranked keyword search as the example of one application of our generic construction. The notable feature of our proposed construction of verifiable ranked keyword search is that it eliminates the interactions between the data owners and users and this is important because we can reduce the communication overhead of ranked keyword search schemes.

TABLE 3
The detail of our proposed verifiable ranked keyword search which is constructed based on our proposed generic construction of VABKS.

Verifiable Ranked Keyword Search Using Proposed Generic Construction of VABKS

- **Setup:**
  1. The TA runs setup algorithm $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{Setup}(\mathsf{n})$ to generate public parameter pp and master secret key msk and broadcasts pp. It also selects the symmetric encryption algorithm $\pi : \{C \leftarrow \mathsf{Enc}_\pi(m, K), m := \mathsf{Dec}_\pi(C, K)\}$.
  2. The TA runs algorithm $\mathsf{sk}_{U_i} \leftarrow \mathsf{KeyGen}(I_{\mathsf{KeyGen}}, \mathsf{msk})$ and sends users U_i secret key sk_{U_i} through a secure and authenticated channel.
  3. The data owner extracts keyword group $KG = \{\omega_1, \ldots, \omega_n\}$ from document set $Doc = \{F_1, \ldots, F_m\}$ and for each $\omega_i \in KG$,
     a) builds $\mathcal{F}(\omega_i) = \{F_{ij} : \omega_i \in F_{ij}\}$
     b) for $1 \leq j \leq |\mathcal{F}(\omega_i)|$:
        i) Calculating the score of file F_{ij} which is denoted as $S_{ij} = Q(F_{ij}, \omega_i)$. Function $Q(.,.)$ is used for computing the score of file F_{ij}.
        ii) $S_i = \{(S_{i1}, id(F_{ij})), \ldots, (S_{i|\mathcal{F}(\omega_i)|}, id(F_{i|\mathcal{F}(\omega_i)|}))\}$ ($id(F_i)$ is the identifier of file F_i which is defined in Table 1.)
  4. $S := (S_1, \ldots, S_n)$
  5. $(\mathsf{Index}, D_{\mathsf{cph}}) \leftarrow \mathsf{RankBuildIndex}(\mathsf{pp}, (KG, S), (I_{\mathsf{Enc}}, I_{\mathsf{Enc}'}))$
  6. The data owner encrypts of documents Doc as follows:
     a) $\mathbf{V}'_0 = H_2(I_{\mathsf{Enc}'})$
     b) $K_{\mathsf{EncKey}} \xleftarrow{\$} \{0,1\}^\lambda, C_{K_{\mathsf{Enc}}} \leftarrow \mathsf{BroadEnc}(\mathsf{BE.pp}, \mathbf{V}'_0, K_{\mathsf{Enc}})$
     c) $\forall 1 \leq j \leq m : C_{F_j} = \mathsf{Enc}_\pi(F_j, K_{\mathsf{Enc}})$
     d) $\mathsf{DocCph} := (C_{K_{\mathsf{Enc}}}, C_{F_1}, \ldots, C_{F_m})$
  7. Outsources searchable ciphertext $\big((\mathsf{Index}, D_{\mathsf{cph}}), \mathsf{DocCph}\big)$
- **Retrieval:**
  1. $st_{\omega^*,j} \leftarrow \mathsf{TokenGen}(\mathsf{SK}_{U_j}, \omega^*)$
  2. Data user U_j sends the cloud generated search token $st_{\omega^*,j}$
  3. The cloud runs search algorithm $\{\mathsf{rslt}, \mathsf{proof}\} \leftarrow \mathsf{SearchIndex}(st_{\omega^*,j}, (\mathsf{Index}, D_{\mathsf{cph}}))$
  4. The data user verifies the correctness of search result by running $\{0,1\} := \mathsf{Verify}(SK_j, \omega^*, st_{\omega^*,j}, \mathsf{rslt}, \mathsf{proof})$
  5. If the search result is valid and documents which contain ω^* have found, then $\mathsf{rslt} = (C_{\omega^*} = C_{\omega^*} || C_{S^*}, R)$ and data user acts as follows:
     a) $S^* := \mathsf{BroadDec}(\mathsf{BE.pp}, C_{S^*}, \mathsf{BE.sk}_j)$
     b) Request to download the top-k relevant files with identifiers $id(F_{i_1}), \ldots, id(F_{i_k})$
     c) The cloud finds the encryption of files corresponding to identifiers $id(F_{i_1}), \ldots, id(F_{i_k})$ i.e., $(C_{K_{\mathsf{Enc}}}, C_{F_{i_1}}, \ldots, C_{F_{i_k}})$
  6. The data user who satisfies the access control policy can decrypt $(C_{K_{\mathsf{Enc}}}, C_{F_{i_1}}, \ldots, C_{F_{i_k}})$ as follows:
     a) $K_{\mathsf{Enc}} := \mathsf{BroadDec}(\mathsf{BE.pp}, C_{K_{\mathsf{Enc}}}, \mathsf{BE.sk}_j)$
     b) $\forall 1 \leq j \leq k : F_{i_1} := \mathsf{Dec}_\pi(C_{F_{i_1}}, K_{\mathsf{Enc}})$

TABLE 4
The comparison of the existing ranked keyword search schemes.

References	No **interaction** between data user and data owner	**Verifiability**	**Multi keywords search**	**Fuzzy search**	**Exact search**	**Semantic search**	**Provable security**
[37]	✗	✗	✓	✓	✗	✓	✗
[32]	✗	✗	✗	✗	✓	✗	✗
[36]	✗	✓	✓	✓	✗	✗	✓
[35]	✗	✗	✓	✓	✗	✗	✗
[39]	✗	✗	✓	✓	✗	✗	✗
[34]	✗	✗	✓	✓	✗	✗	✗
[38]	✗	✓	✓	✓	✗	✓	✓
Ours	✓	✓	✗	✗	✓	✗	✓

References

[1] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000*. IEEE Computer Society, 2000, pp. 44–55.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*. Springer, 2004, pp. 506–522.

[3] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*. Springer, 2001, pp. 213–229.

[4] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," *J. Cryptology*, vol. 21, no. 3, pp. 350–391, 2008.

[5] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*. Springer, 2006, pp. 290–307.

[6] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in *Computational Science and Its Applications - ICCSA 2008, International Conference, Perugia, Italy, June 30 - July 3, 2008, Proceedings, Part I*. Springer, 2008, pp. 1249–1259.

[7] J. W. Byun, H. S. Rhee, H. Park, and D. H. Lee, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in *Secure Data Management, Third VLDB Workshop, SDM 2006, Seoul, Korea, September 10-11, 2006, Proceedings*. Springer, 2006, pp. 75–83.

[8] Q. Tang and L. Chen, "Public-key encryption with registered keyword search," in *Public Key Infrastructures, Services and Applications - 6th European Workshop, EuroPKI 2009, Pisa, Italy, September 10-11, 2009, Revised Selected Papers*. Springer, 2009, pp. 163–178.

[9] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA*. IEEE, 2010, pp. 441–445.

[10] M. Nishioka, "Perfect keyword privacy in PEKS systems," in *Provable Security - 6th International Conference, ProvSec 2012, Chengdu, China, September 26-28, 2012. Proceedings*. Springer, 2012, pp. 175–192.

[11] P. Xu, H. Jin, Q. Wu, and W. Wang, "Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack," *Computers, IEEE Transactions on*, vol. 62, no. 11, pp. 2266–2277, 2013.

[12] M. Zhang, X. A. Wang, X. Yang, and W. Cai, "Efficient predicate encryption supporting construction of fine-grained searchable encryption," in *2013 5th International Conference on Intelligent Networking and Collaborative Systems, Xi'an city, Shaanxi province, China, September 9-11, 2013*. IEEE, 2013, pp. 438–442.

[13] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," *J. Cryptology*, vol. 26, no. 2, pp. 191–224, 2013.

[14] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*. ACM, 2006, pp. 89–98.

[15] F. Han, J. Qin, H. Zhao, and J. Hu, "A general transformation from KP-ABE to searchable encryption," *Future Generation Comp. Syst.*, vol. 30, pp. 107–115, 2014.

[16] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: verifiable attribute-based keyword search over outsourced encrypted data," in *2014 IEEE Conference on Computer Communications, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014*. IEEE, 2014, pp. 522–530.

[17] Y. Desmedt, "Verifier-designated signatures," vol. 3, 2003.

[18] F. Laguillaumie, B. Libert, and J.-J. Quisquater, "Universal designated verifier signatures without random oracles or non-black box assumptions," in *Security and Cryptography for Networks*. Springer, 2006, pp. 63–77.

[19] W. Liu, J. Liu, Q. Wu, and B. Qin, "Hierarchical identity-based broadcast encryption," in *Information Security and Privacy - 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings*. Springer, 2014, pp. 242–257.

[20] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[21] J. Horwitz and B. Lynn, "Toward hierarchical identity-based encryption," in *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*. Springer, 2002, pp. 466–481.

[22] C. Gentry and A. Silverberg, "Hierarchical id-based cryptography," in *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*. Springer, 2002, pp. 548–566.

[23] D. Boneh and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles," in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*. Springer, 2004, pp. 223–238.

[24] D. Boneh, X. Boyen, and E. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*. Springer, 2005, pp. 440–456.

[25] C. Gentry and S. Halevi, "Hierarchical identity based encryption with polynomially many levels," in *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*. Springer, 2009, pp. 437–456.

[26] W. Liu, J. Liu, Q. Wu, B. Qin, and Y. Li, "Practical chosen-ciphertext secure hierarchical identity-based broadcast encryption," *International Journal of Information Security*, pp. 1–16, 2015.

[27] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Springer, 1996, pp. 143–154.

[28] S. Saeednia, S. Kremer, and O. Markowitch, "An efficient strong designated verifier signature scheme," in *Information Security and Cryptology - ICISC 2003, 6th International Conference, Seoul, Korea, November 27-28, 2003, Revised Papers*. Springer, 2003, pp. 40–54.

[29] F. Laguillaumie and D. Vergnaud, "Designated verifier signatures: Anonymity and efficient construction from any bilinear map," in *Security in Communication Networks, 4th International Conference, SCN 2004, Amalfi, Italy, September 8-10, 2004, Revised Selected Papers*. Springer, 2004, pp. 105–119.

[30] S. S. M. Chow, "Identity-based strong multi-designated verifiers signatures," in *Public Key Infrastructure, Third European PKI Workshop: Theory and Practice, EuroPKI 2006, Turin, Italy, June 19-20, 2006, Proceedings*. Springer, 2006, pp. 257–259.

[31] F. Laguillaumie and D. Vergnaud, "Multi-designated verifiers signatures: anonymity without encryption," *Inf. Process. Lett.*, vol. 102, no. 2-3, pp. 127–132, 2007.

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

[33] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," in *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10-15 April 2011, Shanghai, China*. IEEE, 2011, pp. 829–837.

[34] ——, "Privacy preserving multi-keyword ranked search over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 222–233, 2014.

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

[36] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 3025–3035, 2014.

[37] Z. Fu, X. Sun, N. Linge, and L. Zhou, "Achieving effective cloud search services: multi-keyword ranked search over encrypted cloud data supporting synonym query," *IEEE Trans. Consumer Electronics*, vol. 60, no. 1, pp. 164–172, 2014.

[38] C. Chen, X. Zhu, P. Shen, J. Hu, S. Guo, Z. Tari, and A. Zomaya, "An efficient privacy-preserving ranked keyword search method," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[39] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.