IJICTR
International Journal of Information & Communication Technology Research

Volume 10- Number 1 – Winter 2018 (48-55)

# *A Secure Attribute-Based Keyword Search Scheme Against Keyword Guessing And Chosen Keyword Attacks*

Vahid Yousefipoor
School of Electrical Engineering
Sharif University of Technology
Tehran, Iran
yousefipoor_vahid@ee.sharif.edu

Mohammad Hassan Ameri
Electronics Research Institute
Sharif University of Technology
Tehran, Iran
ameri_mohammadhasan@ee.sharif.edu

Javad Mohajeri
Electronics Research Institute
Sharif University of Technology
Tehran, Iran
mohajer@sharif.ir

Taraneh Eghlidos*
Electronics Research Institute
Sharif University of Technology
Tehran, Iran
teghlidos@sharif.ir

Received: 30 October, 2017 - Accepted: 19 February, 2018

*Abstract*—**To provide the privacy of the users who receive some computing services from the cloud, the users must encrypt their documents before outsourcing them to the cloud. Computation on outsourced encrypted data in the cloud rises some complexity to the system especially in the case when an entity would like to find some documents related to a special keyword. Searchable encryption is a tool for data owners to encrypt their data in a searchable manner. Generally, there exist two kinds of searchable encryption, namely symmetric (secret key) and asymmetric (public key) ones. Most of the public key searchable encryption schemes are vulnerable to the keyword guessing attack (KGA). In this paper, we propose an attribute-based keyword search scheme which is proved to be secure against KGA. Also, we formally prove that the proposed scheme is secure against another attack, namely the chosen keyword attack (CKA) in the random oracle model.**

***Keywords-Attribute-based keyword search, searchable encryption, keyword guessing attack, chosen keyword attack, cloud security***

## I. INTRODUCTION

Nowadays, because of the promotions and the developments of Information technology (IT) and digital communications systems and the need for powerful resources for computation and storage, we are the witness of migration from existing computing schemes to the cloud computing and cloud storage environments. As a result, we can see that the clients and the IT users are the beneficiaries of the services

* Corresponding Author

which are provided by the existing cloud providers like DropBox and etc. In this case, an entity who requests to receive a special service from the cloud provider, it should give the required documents to the cloud server and then receives the result of computations on the sent documents.

However, the cloud providers are not usually fully trusted and we need to protect the privacy of our sensitive documents. In this case, the only way to protect our data is to use cryptographic primitives to encrypt them before outsourcing to the cloud. The encryption process should be done in such a way to allow the cloud server to run the required computations on the stored ciphertext without any problem. For example, by using Fully-Homomorphic Encryption (FHE), the cloud server runs the computations on the ciphertexts and the output is the encrypted form of the results. We recall that the first FHE scheme is proposed by Gentry [1].

One of the most interesting topics for preserving the privacy of the users in the cloud-based environments is searchable encryption. By means of this primitive, the data owners generate a search token related to the intended keyword and then sends it to the cloud. The cloud receives the search token and looks for the required documents by using the received token and sends back the results to the data user without inferring any information about the intended keyword. The existing searchable encryption schemes are divided into two main groups which are symmetric (secret key) searchable encryption and asymmetric (public key) searchable encryption. Song proposed symmetric searchable encryption for the first time [2]. The public key variant of searchable encryption was proposed by Bone the so-called public key encryption with keyword search (PEKS) [3].

Based on the different public key cryptographic primitives, various kinds of public key searchable encryption have been proposed in the literature, [4], [5], [6] and [7]. In PEKS, all the data owners know the public key of the data users and encrypt the keyword in a searchable manner by means of the public key of the intended data user and outsource the resulting ciphertext to the cloud. As the data user knows the secret key, he is the lone authorized entity to generate a search token for a keyword and then sends it to the cloud. The cloud receives the search trapdoor and runs the search algorithm to find the related document to the queried keyword.

In this paper, we study the case in which the data users and the data owners are associated with a set of attributes and they receive their secret keys from a Trusted Authority (TA), corresponding to their attributes. So, the data owners encrypt their documents based on a search control policy and outsource the resulting ciphertext to the cloud server. Then, the data users whose attributes satisfies the intended search control policy a are the sole entities who can generate a valid search token. In this model, the data users are able to generate the required search token without any interactions with the data owners [8]. This kind of searchable encryption is called attribute-based keyword search scheme (ABKS) which is introduced by Zeng et al. [8]. Their scheme is presented with the inspiration of attribute-based encryption (ABE) [9] and the two proposed variants called ciphertext policy ABKS (CP-ABKS) and key-policy ABKS (KP-ABKS). In KP-ABKS the cryptographic credentials are associated with the search control policy and in CP-ABKS the ciphertext of the keyword is associated with the search control policy [8]. Some ABKS scheme was proposed in [10, 11].

### *A. Our contribution*

In the public key searchable encryption, the cloud providers always receive some search tokens from different authorities and each of them is generated for a special keyword. It may happen that the cloud tries to infer some information from the received search tokens and realizes the corresponding keyword. Actually, the cloud applies the keyword guessing attack (KGA). Byun, for the first time, proposed KGA against the public key searchable encryption schemes [12]. Usually, the set of the keywords are limited and as the adversary knows the public key, it starts to encrypt all the possible keywords. Then it uses the existing token and runs the search algorithm to find the corresponding ciphertext. As the adversary knows the related keyword, it can realize the associated keyword to the mentioned search token. Motivated by this attack, we try to propose an ABKS scheme which is secure against KGA.

- In our design, the data owner generates a fuzzy and the exact ciphertext associated to the intended keyword. The data user generates the fuzzy and exact keyword search and sends the fuzzy search token to the cloud server and keeps the exact keyword search secret. In this scheme, it may happen that the cloud finds two or more ciphertexts associated to the different keywords of intended keyword (the results can also include the ciphertext of intended keyword) using the fuzzy search token and then sends the results to the data user. The data user uses the exact search token and runs the search operation on the received ciphertexts from cloud to find the right documents associated to the considered keyword. As the cloud receives a fuzzy search token it cannot find the related keyword. As a result, the proposed scheme takes advantage of being secure against KGA.
- Another common attack in the cloud environment is chosen keyword attack [8]. In this attack, the adversary has a valid encrypted keyword and aims to know the corresponding keyword. This attack is similar to chosen plaintext attack against cryptographic schemes. We show that the proposed scheme is secure against the chosen keyword attack in the random oracle model.
- In addition, the performance analysis and comparison results shows the practical and deployable aspects of the proposed scheme which prove that our proposal is practical in real world applications.

The rest of this paper is organized as follows. In section 2 we review some preliminaries to introduce our scheme. Section 3 is devoted the introduction of both generic and concrete constructions of our scheme. Section 4 analyses its security properties. In section 5 we discuss the performance of the proposed scheme. Finally, we conclude the paper in section 6.

## II. PRELIMINARIES

### A. Bilinear Map and Access Structure

• Bilinear Maps [13]: Let $g$ be a generator of a cyclic group $G$ of prime order $p$ and $G_T$ be a cyclic group with the same order. A mapping $\hat{e}: G \times G \to G_T$ is a bilinear map if for all $a, b \in \mathbb{Z}_p$ the following conditions hold: 1) $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$ 2) $\hat{e}(g, g) \neq 1$ 3) $\hat{e}(g, g)$ is efficiently computable.

• Access structure: It is usually constructed using a concept called access tree [14]. An access tree has two types of nodes: a 'leaf' and a 'threshold gate'. Each node is denoted by $v$. Each leaf is associated with an attribute which denoted by $att(v)$ and each threshold gate has a threshold value. The set of all leaves of the access tree $T$ is denoted by $lv(T)$. Finally, a subtree of $T$ with the root node $v$ denoted by $T_v$. For creating a desired access structure, we use two probabilistic polynomial time (PPT) algorithms, namely $Dist$ and $Recon$. $\{q_v(0) \mid v \in lv(T)\} \leftarrow Dist(T, s)$ distributes the secret value $s$ according to $T$ and generates the polynomial $q_v(0)$ of degree $k_v - 1$. Each leaf node has $k_v = 1$ and for threshold gate $k_v = 1$ or $k_v =$ the number of children of a node $v$, the former denotes OR gate and the latter denotes AND gate. The $Recon$ is the inverse PPT algorithm for $Dist$. Assume that $v_1, ..., v_m$ are the leaves of $T$. For each $j \in \{1, ..., m\}$ and $g, h \in G$ $\Delta_{v_j} = \hat{e}(g, h)^{q_{v_j}(0)}$. By running the algorithm $\hat{e}(g, h)^s \leftarrow Recon\{T, (\Delta_{v_1}, ..., \Delta_{v_m})\}$ we can reconstruct the secret value $s$.

## III. PROPOSED SCHEME

In this section, we first explain system model of our scheme and then give its both generic and concrete constructions.

### A. System model

The framework of our proposed scheme involves four entities that are shown in Fig. 1.

• Cloud Server (CS): It has the strong computational capability and very large storage capacity. It stores and processes data and provides users with keyword search service.

• Data Owner (DO): This entity wants to share his data. For this propose, he encrypts the desired data under certain access policy and finally outsources them to the CS.

• Data User (DU): This entity searches for certain keywords on the stored data in the CS. Using his secret key the DU generates search token and fuzzy search token and sends the latter to the CS for keyword searching. When the CS returns search results to the DU, the DU uses search token for keyword searching on receiving data.

• Trusted Authority (TA): The fully trusted third party manages a set of all attributes and distributes them between DOs and DUs. Also, TA generates secret keys for data users according to their attributes.

### B. Generic construction

We explain the proposed scheme in five stages. Stages 1 and 2 have the same structures in the standard ABKS model [8], but we observe some differences with the standard model in stages 3 and 4. In addition, the proposed scheme has new stages, stage 5, that does not exist in the standard model.

• Stage 1: In this stage, TA calls the $Setup$ algorithm to generate public parameter $pm$ and the master secret key $mk$. Then DOs and DUs register on the system and get their corresponding set of attributes $Att$ and $Att'$, respectively. In addition, for each DU, TA calls the $KeyGen$ algorithm to generate a secret key $sk$. This key is used to produce the tokens used when performing the keyword search.

• Stage 2: In this stage, the DO stores his file $F$ containing a certain keyword $kw$ to CS. He calls $Enc$ algorithm to encrypt $F$ using $kw$ and his access tree $T$. Thus, the ciphertext $C$ is generated. $Enc$ is based on the CP-ABE. Each DU satisfying access tree $T$, can access to the file $F$.

• Stage 3: In this stage, the DU wants to search for all files containing a keyword $kw$. He calls $TokenGen$ algorithm to make the search token $tk$ and the fuzzy search token $ftk$. He queries $ftk$ to CS and keeps $tk$ secret.

• Stage 4: In this stage, the CS receives the fuzzy search token $ftk$ from a DU. Then it calls the $FuzzySearch$ algorithm to find all files that match $ftk$. Finally, CS sends the search results $SR$ to the DU. The $SR$ contains all files with the desired keyword $kw$ and another keyword.

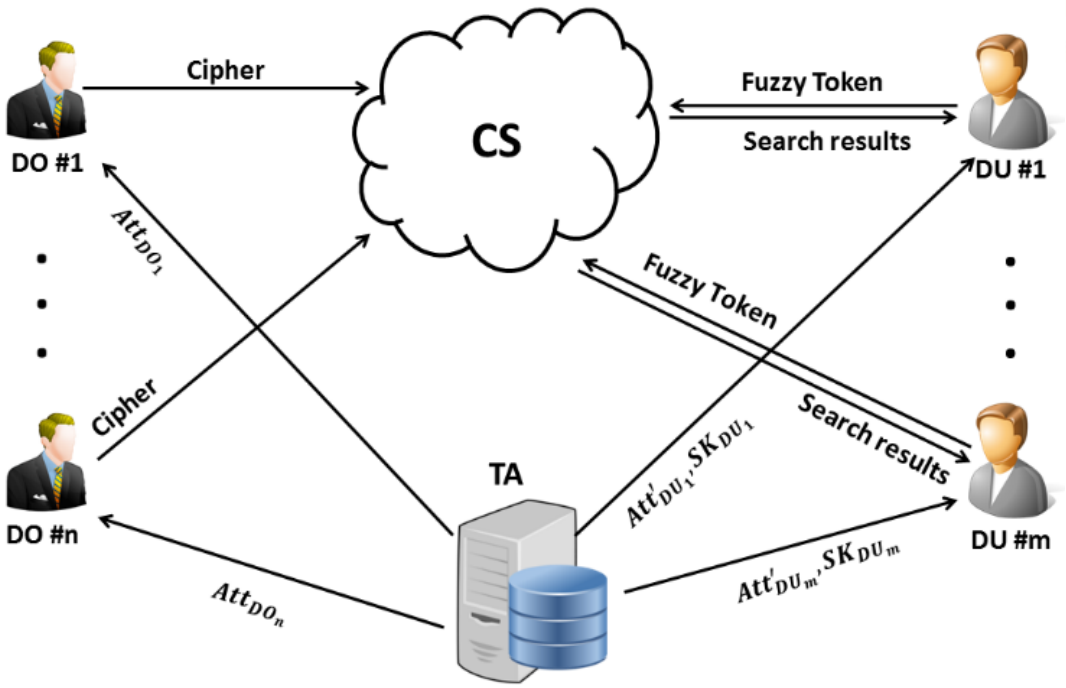


*Fig. 1. System model architecture*

• Stage 5: In this stage, the DU executes $Search$ algorithm on $SR$ and extracts all files that contain the keyword $kw$

*C. Concrete construction*

In this part, we describe all algorithms that have used in the generic construction.

$Setup(1^n) \rightarrow (pm, mk)$: This algorithm is run by TA. It takes as input the security parameter $n$ and generates the public parameter $pm$ and the master secret key $mk$. The TA selects a bilinear map $\hat{e}: G \times G \rightarrow G_T$, where both $G$ and $G_T$ are cyclic groups of order $p$. Let $H_1: \{0,1\}^* \rightarrow G$ and $H_2: \{0,1\}^* \rightarrow \mathbb{Z}_p$ be two hash functions. The TA also selects randomly $x, y, z \in \mathbb{Z}_p$ and $g \in G$. The TA defines $Fuzz$ function [7] for the set of all keywords $KW = \{kw_1, ...kw_t\}$ as follows:

- if $|KW|$ is even, $Fuzz(kw_i)$ outputs:

$$Fuzz(kw_i) = \begin{cases} (kw_{i-1} \| kw_i) & i\ even \\ (kw_i \| kw_{i+1}) & i\ odd \end{cases} \quad (1)$$

- if $|KW|$ is odd, $Fuzz(kw_i)$ outputs:

$$Fuzz(kw_i) = \begin{cases} (kw_{i-1} \| kw_i) & i\ even \\ (kw_i \| kw_{i+1}) & i\ odd \\ (kw_{|KW|-2} \| kw_{|KW|-1} \| kw_{|KW|}) & i \geq |KW| - 2 \end{cases} \quad (2)$$

Finally the TA sets:

$$pm = (H_1, H_2, g, p, G, G_T, \hat{e}, Fuzz, g^x, g^y, g^z) \quad (3)$$

$$mk = (x, y, z) \quad (4)$$

$KeyGen(mk, Att') \rightarrow sk$: This algorithm is run by the TA to generate the secret key $sk$. The authorized DU gives TA his set of attributes $Att'$ and gets his secret key $sk$. This algorithm selects randomly $b$ in $\mathbb{Z}_p$ and sets. Then for each $at_j \in Att'$ it selects randomly $b_j \in \mathbb{Z}_p$ and computes $R_j = g^b H_1(at_j)^{b_j}, U_j = g^{b_j}$ and sets:

$$sk = (Att', R, \{(R_j, U_j) \mid at_j \in Att'\}) \quad (5)$$

$Enc(kw, T) \rightarrow C$: This algorithm takes as inputs the keyword $kw$ and the DO's access tree $T$ and outputs searchable ciphertext $C$. It selects randomly $b_1, b_2 \in \mathbb{Z}_p$ and sets $I^* = g^{yb_2}, I = g^{zb_1}, I' = g^{x(b_1+b_2)} g^{yH_2(kw)b_1}$ and $I'_{fuzz} = g^{x(b_1+b_2)} g^{yH_2(Fuzz(kw))b_1}$. For each $v \in lv(T)$ this algorithm runs $q_v(0) \leftarrow Dist(T, b_2)$ then it computes $I_v = g^{q_v(0)}, O_v = H_1(att(v))^{q_v(0)}$ and sets:

$$C = (T, I, I^*, I', I'_{Fuzz}, \{(I_v, O_v) \mid v \in lv(T)\}) \quad (6)$$

$TokenGen(sk, kw) \rightarrow Token$: This algorithm generates the DU's search tokens $Token = (tk, ftk)$ to search for files containing the keyword $kw$. The DU randomly selects $c \in \mathbb{Z}_p$ and computes $t_1 = (g^x g^{yH_2(kw)})^c, t_{1,fuzz} = (g^x g^{yH_2(Fuzz(kw))})^c$,

$t_2 = g^{zc}$ and For each $at_j \in Att'$ the algorithm uses $sk$ and computes $R_j^* = R_j^c$, $U_j^* = U_j^c$ and sets:

$$tk = (Att', t_1, t_2, t_3, \{(R_j^*, U_j^*) \mid at_j \in Att'\}) \quad (7)$$

$$ftk = (Att', t_{1,fuzz}, t_2, t_3, \{(R_j^*, U_j^*) \mid at_j \in Att'\}) \quad (8)$$

$$Token = (tk, ftk) \quad (9)$$

$FuzzySearch(ftk, C) \rightarrow SR$: The CS uses the fuzzy token $ftk$ and executes this algorithm on all files that have been stored in the cloud. It searches for an attribute set $S$ satisfying the access tree $T$. If $S$ does not exist, the search process is stopped and the CS returns 0 to the DU. Else, for each $at_j \in S$ the CS computes $\Delta_v = \hat{e}(R_j^*, I_v) / \hat{e}(U_j^*, O_v) = \hat{e}(g,g)^{bcb_2}$ ( $att(v) = at_j$ for every $v \in lv(T)$ ) and runs $\hat{e}(g,g)^{cbq_{root}(0)} \leftarrow Recon(T, (\Delta_v)_{v \in S})$ so that. Now the CS checks equation (10) for all files stored in the cloud and sends those files satisfying this equation as the search results $SR$ to the DU.

$$\hat{e}(I'_{fuzz}, t_2) = \hat{e}(I, t_{1,fuzz}) \Delta_{root} \hat{e}(I^*, t_3) \quad (10)$$

The equation (11) is correct because:

$$\begin{aligned} \hat{e}(I'_{fuzz}, t_2) &= \hat{e}(g^{x(b_3+b_2)} g^{yFuzz(kw)b_1}, g^{zc}) \\ &= \hat{e}(g,g)^{xyzb_1 cFuzz(kw)(b_3+b_2)} \\ &= \hat{e}(I, t_{1,fuzz}) \Delta_{root} \hat{e}(I^*, t_3) \end{aligned}$$

$Search(tk, SR) \rightarrow D$: This algorithm is run by the DU which is similar to $FuzzySearch$ algorithm. The DU extracts the files $D$ containing the keyword $kw$ from $SR$. The DU checks the following equation on all files in $SR$ and saves the files satisfying this equation.

$$\hat{e}(I', t_2) = \hat{e}(I, t_1) \Delta_{root} \hat{e}(I^*, t_3) \quad (11)$$

## IV. SECURITY ANALYSIS

In this section, we analyze the security of the proposed scheme against keyword guessing and chosen keyword attack.

### A. Security against keyword guessing attack

Assume a probabilistic polynomial-time (PPT) adversary $\mathrm{A}$, who may be an unauthorized DU. In this attack $\mathrm{A}$ has a valid search token and he knows the set of all keywords. He wants to find a keyword corresponding to the search token. The adversary runs the following algorithm for each keyword:

1. $\mathrm{A}$ encrypts the keyword and generates a keyword searchable ciphertext and then uploads the ciphertext to the cloud.
2. $\mathrm{A}$ sends the valid search token to the CS.
3. The CS sends search results to $\mathrm{A}$. if the search results contain the ciphertext, $\mathrm{A}$ returns the keyword.

In the most previous ABKS schemes, $\mathrm{A}$ can easily runs the above algorithm and find the correct keyword with high probability [8], because in these schemes a search token corresponds to a special keyword the algorithm only outputs the encrypted keyword.. So the adversary ensures that KGA algorithm outputs the correct keyword. In the proposed scheme, we have solved this problem using $Fuzz$ function. Assume that the adversary $\mathrm{A}$ has a valid fuzzy search token $ftk$ and knows the set of all keywords $KW = \{kw_1, ..., kw_t\}$. He implements the KGA algorithm as follows:

1. Set i=1
2. Generates the corresponding ciphertext $C$ to the keyword $kw_i$, using the desired access tree $T$ and runs $Enc(kw_i, T)$. Then he outsources $C$ to the CS.
3. $\mathrm{A}$ sends $ftk$ to the CS.
4. The CS sends search results $SR$ to $\mathrm{A}$. if $C \in SR$, then $\mathrm{A}$ returns $kw_i$, else $i = i+1$ and then returns to step 1. if $i = |KW|$ and $C \notin SR$ then he returns $\perp$.

Because the $ftk$ is valid, an adversary $\mathrm{A}$ never returns $\perp$. Assume that for an index $j \in \{1, ..., t\}$ an adversary $\mathrm{A}$ returns $kw_j$. Then according to the definitions of $Fuzz$ he also returns $kw_{j-1}$ or $kw_{j+1}$. Now for making successful attacks, $\mathrm{A}$ must correctly guess between $j$ and $j-1$ (or $j+1$ ). Assume that $guess$ denotes an occurrence of a true guessing between $j$ and $j-1$ (or $j+1$ ) and $success$ denotes the occurrence of an adversary success occurrence.So the probability of success in this attack for $\mathrm{A}$ is:

$$\Pr(success) = \Pr(guess) \quad (12)$$

According to the definition of $Fuzz$, $\mathrm{A}$ has no information about the true value of the index, so $\Pr\{Success\} \leq 1/2 + negl(n)$ that $negl(.)$ is a negligible function and the proposed scheme is secure against KGA.

### B. Security against chosen keyword attack

In chosen keyword attack, a probabilistic polynomial time (PPT) adversary $\mathrm{A}$ has a valid ciphertext $C$ and aims to find the corresponding keyword. $\mathrm{A}$ can access to two random oracles, namely $O_{KeyGen}(Att)$ and $O_{TokenGen}(Att, kw)$. An adversary queries limited numbers of different set of attributes $Att$ and gets the corresponding secret key

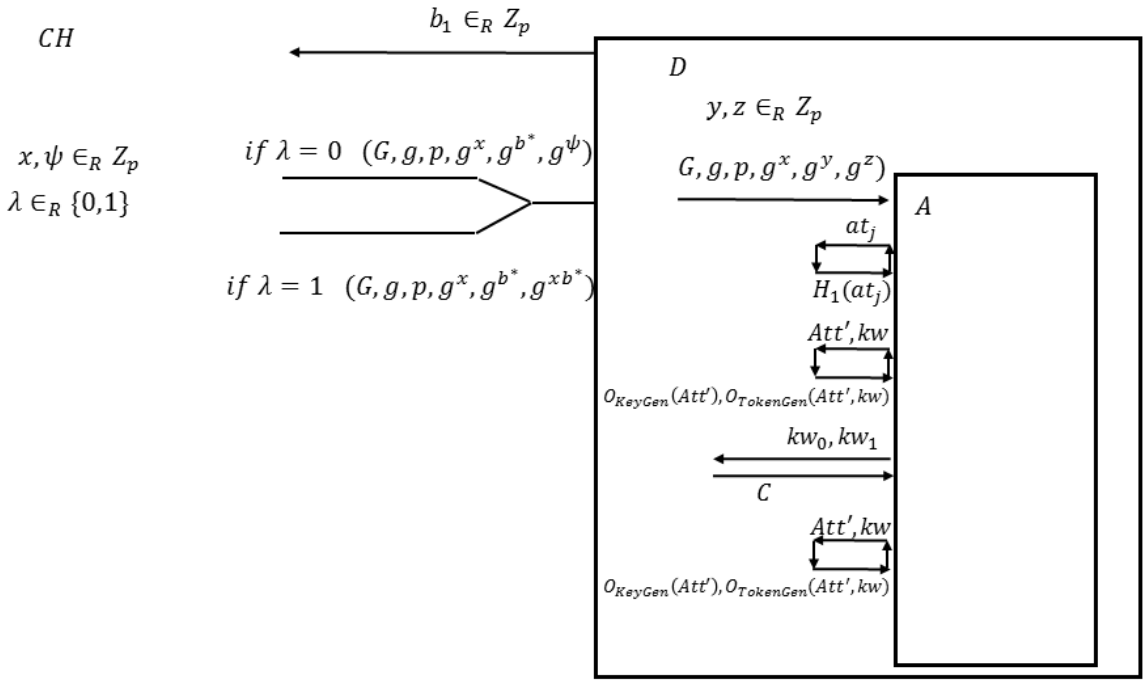


*Fig. 2. The game for proofing security of the proposed scheme against CK*

$sk$ . With accessing to $O_{TokenGen}(Att, kw)$ , an adversary can query set of attributes $Att$ with a keyword $kw$ and gets the related token [8]. We show the proposed scheme is secure against this attack. The security of the proposed scheme is reduced to the hardness of solving the decisional diffie-hellman (DDH) problem [15] for every PPT distinguisher.

Assume that $G$ is a cyclic group of order $p$ and $g \in_R G$ and $x, y, z \in_R Z_p$ . In DDH problem we assume distinguishing between two six-tuples $(G, g, p, g^x, g^y, g^{xy})$ and $(G, g, p, g^x, g^y, g^z)$ is difficult for any PPT distinguisher. Our proof is based on the difficulty of the DDH problem and another assumption that $H_1$ is modeled as a random oracle. Structure of the our proof is consists of three entities, a PPT adversary $\mathrm{A}$ that wants to do chosen keyword attack to the proposed scheme, a PPT adversary $D$ that wants to solve DDH problem and a challenger $CH$ . First $D$ chooses $b_1 \in_R Z_p$ and sends it to the $CH$ . Then $CH$ selects $x, b_2, \psi \in_R Z_p$ , a cyclic group $G$ of order big prime number $p$ and $g \in_R G$ and then $CH$ computes $b^* = b_1 + b_2$ . Now $CH$ chooses $\lambda \in \{0,1\}$ randomly. If $\lambda = 0$ then he sets $Q = g^{\psi}$ otherwise, set $Q = g^{xb^*}$ , Then he sends $(G, g, p, g^x, g^{b^*}, Q)$ to $D$ . $D$ selects $y, z \in_R Z_p$ and sends six-tuple $(G, g, p, g^x, g^y, g^z)$ to $\mathrm{A}$ . $D$ must model the $H_1$ as a random oracle. For this reason, $D$ creates a hash table as a Tabel. 1. For every query $at_j$ which is received from $\mathrm{A}$ first, $D$ chooses $\rho_j \in_R Z_p$ and sends to him $g^{\rho_j}$ . Also, $D$ addes the corresponding row with computed values. $D$ models $O_{KeyGen}(.)$ and $O_{TokenGen}(.)$ for $\mathrm{A}$ . When $\mathrm{A}$ queries $Att'$ for $O_{KeyGen}(.)$ , $D$ selects $b \in_R Z_p$ and calculates $R = g^{(xz-b)y^{-1}}$ . Note that for computing $R = g^{(xz-b)y^{-1}}$ , $D$ does not need to know $x$ , because he knows $y, z, b, g^x$ and he computes $R = [(g^x)^z . g^{-b}]^{y^{-1}}$ . Then for every $at_j \in Att'$ , $D$ chooses $b_j \in_R Z_p$ and computes $R_j = g^b H_1(at_j)^{b_j}$ and $U_j = g^{b_j}$ by using the hash table. Finally, $D$ outputs $sk = (Att', R, \{(R_j, U_j) \mid at_j \in Att'\})$ to $\mathrm{A}$ . If $\mathrm{A}$ queries $(Att', kw)$ for $O_{TokenGen}(.)$ , $D$ calls $O_{KeyGen}(Att')$ first and executes the algorithm $TokenGen(sk, kw) \rightarrow Token$ and returns $Token$ to $\mathrm{A}$ . Also, $D$ saves all $Att'$ which are matched with $T_A$ , that is the access tree of the adversary $\mathrm{A}$ . N $\mathrm{A}$ sends two challenge keywords $kw_0, kw_1$ to $D$ . If these keywords do not match with $T_A$ , $D$ uses the value $b_1$ , which he had calculated first, and calls the algorithm $Dist(T_A, b_1) \rightarrow \{q_v(0) \mid v \in lv(T_A)\}$ . Then $D$ uses $(G, g, p, g^x, g^{b^*}, Q)$ and calculates $I' = Q$ , $I = g^{zb_2}$ , $I^* = g^{yb_1}$ . Then for each $v \in T_A$

*Table.1.The hash table*

| $at_j$ | $\rho_j$ | $H_1(at_j)$ |
|---|---|---|
| $at_1$ | $\rho_1$ | $H_1(at_1)$ |
| $at_2$ | $\rho_2$ | $H_1(at_2)$ |
| … | … | … |
| $at_{p(l)}$ | $\rho_{p(l)}$ | $H_1(at_{p(l)})$ |

he calculates $O_v = H_1(att(v))^{q_v(0)}$ and $I_v = g^{q_v(0)}$ using hash table. Finally, $D$ returns encrypted keyword $C = (T, I, I^*, I', \{(I_v, O_v) \mid v \in lv(T)\})$ to $A$ . Note that if $Q = g^{xb^*}$ then $C$ is a valid ciphertext based on the proposed scheme. Fig. 2 shows a schematic view of the proof procedure.

Based on the above security game, $A$ can do successful chosen keyword attack against the proposed scheme if $Q = g^{xb^*}$ . When $A$ do this, $D$ find out $Q = g^{xb^*}$ , so $D$ solves DDH problem. But we assume that solving DDH problem if hard for PPT adversary $D$ . As a result doing CKA againt the proposed scheme is hard for $A$ and the proposed scheme is secure against CKA.

## V. PERFORMANCE ANALYSIS

In the proposed scheme, the probability of the adversary's success in KGA is reduced to half compared to the previous schemes. But the computational complexity of the proposed scheme does not get double compared to the standard ABKS [8]. However, we search on the encrypted data twice in the proposed scheme, but the *Search* algorithm is executed only on the small portion of all ciphertexts, namely it is executed on a set of ciphertexts corresponding to two keywords. For numerical comparison with the standard ABKS scheme we were the beneficiaries of the advantages of the experimental results given in [8]. Fig. 3 shows this comparison. This figure shows that if each attribute set has utmost 30 attributes, the searching time of the proposed scheme slightly differs from the standard ABKS scheme. For example if each attribute set has 20 attributes the searching time for the proposed scheme is almost 390 seconds and for standard ABKS is almost 380 seconds. For upper number of attributes, the search complexity of the proposed scheme does not differ tremendously from the standard ABKS.

## VI. CONCLUSION

We have presented a concrete construction for ABKS to design a secure environment for the cloud computing applications. In the proposed scheme, the data owners generate two fuzzy and exact searchable ciphertexts according to a search control policy and send them to the cloud. Then each data user with the authorized set of attributes related to the mentioned search control policy generate two search tokens, called fuzzy and exact, and sends them to the cloud server. The cloud looks for the required documents by using the fuzzy search token and returns the results back to the data user. Finally, the data user runs the search operation with the exact token and finds the exact results. We have shown that the proposed scheme is secure against keyword guessing attack. Also, we proved that the proposed scheme is provable secure in random oracle model and formally prove its security against chosen keyword attack. The security of the proposed scheme is reduced to difficulty of solving the decisional diffie-hellman problem for any probabilistic polynomial time adversary. However, in the proposed scheme, we call search algorithm twice, the scheme has an efficient searching time complexity compared with the standard ABKS scheme.

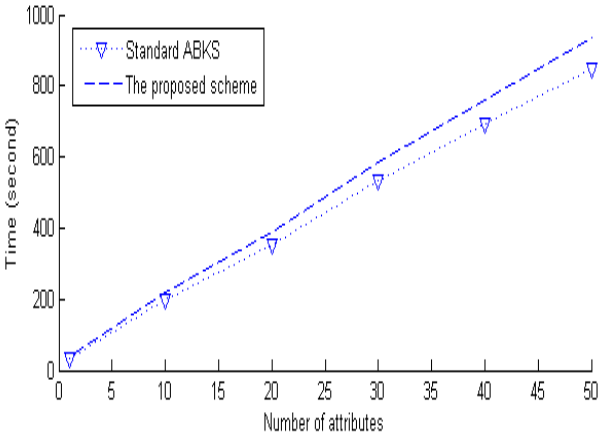


*Fig. 3. Comparison between the search time of the proposed scheme with the standard ABKS*

## REFERENCES

[1] C. Gentry., "Fully homomorphic encryption using ideal lattices." in *STOC*, vol. 9, 2009, pp. 169–178.

[2] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Security and Privacy. S&P Proceedings.IEEE Symposium on*., 2000, pp. 44–55.

[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology Eurocrypt'04*. Springer, 2004, pp. 506–522.

[4] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in *Computational Science and Its Applications–ICCSA 2008*. Springer, 2008, pp. 1249–1259.

[5] Q. Tang and L. Chen, "Public-key encryption with registered keyword search," in *Public Key Infrastructures, Services and Applications*. Springer, 2009, pp. 163–178.

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

[7] 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, , 2013, pp. 2266–2277.

[8] Q. Zheng, S. Xu, and G. Ateniese, "Vabks: Verifiable attribute-based keyword search over outsourced encrypted data," in *INFOCOM, Proceedings IEEE*, 2014, pp. 522–530.

[9] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology Eurocrypt'05*. Springer, 2005, pp. 457–473.

[10] V. Yousefipoor, M. H. Ameri, J. Mohajeri and T. Eghlidos, "Sieving search results for attribute-based keyword search in cloud", *3rd National Conference on Applied Research in Computer Science and Information Technology*, Tehran, 2016.

[11] V. Yousefipoor, M. H. Ameri, J. Mohajeri and T. Eghlidos, "A secure attribute based keyword search scheme against keyword guessing attack" *8-th Interantional Symposium on Telecommunication (IST2016)*, Tehran, 2016.

[12] 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 *Secure Data Management*. Springer, 2006, pp. 75–83.

[13] M. Joye and G. Neven, *Identity-based cryptography*. IOS press, 2009, vol. 2.

[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*, 2006, pp. 89–98.

[15] D. Boneh, "The decisional diffie-hellman problem.", In *Algorithmic number theory*, Springer Berlin Heidelberg, 1998, pp. 48-63.

**Vahid Yousefipoor** is a PhD student in Electrical Engineering at Sharif University of Technology, Tehran, Iran. He received his M.Sc. degree in Electrical Engineering from Sharif University of Technology 2016, and his B.Sc. degree in Electrical Engineering from Amirkabir University of Technology, Tehran, Iran in 2014. His major research interests include Post quantum cryptography, cloud security, functional encryption, provable security and network security. He is also interested in coding theory and its applications in cryptography.

**Mohammad Hassan Ameri** received his B.Sc. degree in Electrical Engineering from Shahid Bahonar University, Kerman, Iran, in 2013 with the honor of first ranked student among the electrical engineering students of the same entrance, and his M.Sc. degree in Electrical Engineering from Sharif University of Technology, Tehran, Iran in 2015. Currently, he is working as a researcher in Electronics Research Institute at Sharif University of Technology. His major research interests include cloud security, Obfuscation, Searchable encryption, provable security, information-theoretic security, network security, design and cryptanalysis of cryptographic protocols.

**Javad Mohajeri** is an assistant professor in Electronics Research Institute and adjunct assistant professor in Electrical Engineering Department at Sharif University of Technology, Tehran, Iran. His research interests include design and cryptanalysis of cryptographic algorithms, and protocols and data security. He is the author/co-author of over 90 research articles in refereed journals/conferences and is one the founding members of Iranian Society of Cryptology.

**Taraneh Eghlidos** received her B.Sc. degree in Mathematics in 1986, from the University of Shahid Beheshti, Tehran, Iran, and the M.Sc. degree in Industrial Mathematics in 1991 from the University of Kaiserslautern, Germany. She received her Ph.D. degree in Mathematics in 2000, from the University of Giessen, Germany. She joined the Sharif University of Technology in 2002 and she is currently an associate professor in the Electronics Research Institute of Sharif University of Technology. Her research interests include interdisciplinary research areas such as symmetric and asymmetric cryptography, application of coding theory in cryptography, and mathematical modelling for representing and solving real world problems. Her current research interests include lattice based cryptography and code based cryptography.