IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 15, 2021, accepted November 11, 2021, date of publication December 16, 2021, date of current version January 21, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3136331

An Encrypted Cloud Email Searching and Filtering Scheme Based on Hidden Policy Ciphertext-Policy Attribute-Based Encryption With Keyword Search

JIAN GAO AND FUCAI ZHOU
Software College, Northeastern University, Shenyang 110819, China

Corresponding author: Fucai Zhou (fczhou@mail.neu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62072090, Grant 61772127, Grant 61472184, and Grant 61872069; in part by the National Science and Technology Major Project under Grant 2013ZX03002006; in part by the Liaoning Province Science and Technology Projects under Grant 2013217004; and in part by the Fundamental Research Funds for the Central Universities under Grant N151704002.

ABSTRACT With the rapid growth of cloud email services, email encryption is beginning to be used more and more to alleviate concerns about cloud privacy and security. However, this increase in usage invites the problem of how to search and filter encrypted emails effectively. Searchable public key encryption is a popular technology to solve encrypted email searching, but encrypted email filtering is still an open problem. We propose an encrypted cloud email searching and filtering scheme based on hidden policy ciphertext-policy attribute-based encryption with keyword search as a new solution. It enables the recipient to search the encrypted cloud email keywords and allows the email filtering server to filter the encrypted email content when receiving the email, as the traditional email keyword filtering service. Our hidden policy scheme is constructed by composite order bilinear groups and proven secure by dual system encryption methodology. Our scheme can be applied to other scenarios such as file searching and filtering and has certain practical value.

INDEX TERMS Attribute-based keyword search, dual system encryption, encrypted email filtering, hidden policy.

I. INTRODUCTION

The total number of business and consumer emails sent and received per day will exceed 319 billion in 2021, and is forecast to grow to over 376 billion by year-end 2025 [1]. Cloud-based email services are seeing rapid growth. The benefits of cloud adoption are clear to all organizations, and an increasing number of organizations, of all sizes, are choosing to migrate to cloud email and collaboration services. Cloud email providers are beginning to provide more security features, such as email encryption, archiving, and other security-related services, which are helping to ease users' concerns about cloud privacy and security.

Email encryption also creates some problems, such as how users are to search for emails without needing bothersome decryption or how the relevant servers are supposed to filter the content of emails (email-related laws in every country or region require the filtering of emails, such as spam, spam containing malicious code, etc.). Moreover, in searching and filtering, cloud servers cannot obtain information about the content of emails. Thus, the main problem we are facing now is how to make it as easy for users to search and filter for encrypted email as it is to search and filter for unencrypted ones in the traditional system. Searchable public key encryption was proposed to address this problem.

Searchable encryption is divided into searchable symmetric encryption and searchable public key encryption. Searchable public key encryption is suitable for encrypted email search scenarios. Boneh *et al.* [3] were the first to put forward the notion of a public key encryption scheme with keyword search (PEKS), which has application in identity-based encryption [2] (IBE) email system. This scheme allows gateways in communication systems to retrieve and determine whether the received email contains keywords to be searched. This solution created the use of searchable encryption technology to solve the searching problem of

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

encrypted email. Subsequently, many PEKS schemes claim to be used in encrypted email searching. Recently, there were some PEKS schemes [29]–[31] for encrypted email. Xu *et al.* [31] proposed an encrypted email multi-keyword search scheme with hidden structures. Li *et al.* [30] proposed a new notion called designated-server identity-based authenticated encryption with keyword search for encrypted emails. Zhang *et al.* [29] proposed a scheme supporting conjunctive keywords search without keyword field. The main security problem of searchable public key encryption is offline keyword guessing attack (KGA) that Byun *et al.* [27] defined for PEKS. Rhee *et al.* [34] proved that the sufficient condition for resisting keyword guessing attack is the indistinguishability of trapdoor. All three schemes [29]–[31] prove the security of keyword trapdoor so that they can resist KGA. However, these schemes did not consider the filtering of encrypted email. Now encrypted email filtering is still an open problem. There have also been some PEKS schemes [4], [32] that have claimed to support encrypted email filtering, but they did not provide a detailed explanation on how to do so.

Boneh *et al.* [5] proposed an abstract and general encrypted email filtering scheme model. Email users could use some partially trusted proxy servers to filter out all encrypted emails recognized as spam according to their own requirements with their scheme. This process achieves the seemingly conflicting goal of hiding the email content on the proxy server and allowing the proxy server to determine whether the email is spam according to the user's own settings. Unfortunately, there are only two paragraphs of text description and a schematic diagram. However, we can see that their scheme model uses searchable encryption to solve this problem. Inspired by this, we came up with a solution. We only need to make the filtering server a particular recipient to solve the problem of encrypted email searching and filtering. Therefore, we decided to design an encrypted email searching and filtering scheme based on attribute-based encryption with keyword search (ABKS).

Our Contribution: Under current circumstances, it would be difficult for an encrypted email to be searched by multiple recipients and filtered by a filter server. However, attribute-based encryption with keyword search, especially ciphertext-policy attribute-based encryption with keyword search (CPABKS), can solve this problem. Therefore, our solution is to design a CPABKS scheme for encrypted cloud email scenarios, aiming to enable both filtering and searching simultaneously. An additional recipient list is specially created when sending an email, and the filter server is added as the recipient. Attributes of users on this recipient list will form a set used as the access structure P of the encrypted keyword index. This way, only the recipients whose attributes meet the control policy can be successfully searched. The filter server can filter keywords within encrypted emails successfully, as the server is added to the additional recipient list. In order to resist KGA and make the scheme obtain full security, the composite order bilinear groups are constructed to realize the policy hiding. According to this direction of thinking, we become the first to propose a hidden policy ciphertext-policy attribute-based encryption with keyword search (HPCPABKS) scheme to solve the problem of encrypted email searching and filtering. Our scheme is not filtering on the email gateway. Like some free email, it only provides recipient server-side filtering. Our scheme has the following advantages:

- Innovatively applies the ABKS design to the encrypted cloud email scenario. The sender creates an additional list of recipients for searching and filtering and adds the recipient filtering server to this list of recipients. The user' attributes in this recipient list are used as the access control policy of the encrypted keyword index. Therefore, the recipients can search keywords by their attributes, and, in turn, the recipient filtering server can filter keywords by its own attributes.
- Our scheme supports copying and grouping multiple emails at the same time and uses only one encrypted keyword index without any additional encrypted index computation costs.
- In our solution, we apply the dual system encryption methodology and the hidden policy to enhance privacy and obtain full security. It can resist KGA and satisfy the confidentiality of the encrypted email system.
- Our scheme has certain practical application value and can further expand anti-virus email protection functions according to malicious email reporting, email attachment processing. It is not only more suitable for encrypted cloud email searching and filtering scenarios, but it can also be extended to encrypted file searching and filtering, as well as other application scenarios.

II. RELATED WORK

The root of HPCPABKS can be traced back to Attribute-based encryption (ABE) proposed by Sahai and Waters [6]. ABE is an extension of traditional public key encryption. The user can express how he or she wants to share the data in the encryption algorithm, make some policies according to the attributes of the receiving user, and share the data according to these policies. The ABE scheme is divided into ciphertext-policy ABE (CPABE) and key-policy ABE (KPABE). In the CP-ABE schemes (e.g. [10]–[12]), the receiver's key is associated with the attribute set, while the ciphertext contains the access policy on the attribute set. Only when the attribute set associated with the receiver's key meets the access policy contained in the ciphertext can it be decrypted. In the KPABE schemes (e.g. [7]–[9]), the ciphertext contains a set of attributes. The key is associated with the attribute set's access policy and can only be decrypted if the attribute set of the ciphertext satisfies the access policy associated with the key.

ABE is an effective solution to realize the privacy of data sharing and security in the cloud platform environment. However, under certain circumstances, in order to not disclose any sensitive information in the access structure, the ABE scheme should support an anonymous access structure to

realize the hidden policy. Nishide *et al.* [16] proposed two CPABE schemes with a hidden access control policy. The user can hide a subset of each attribute's possible values in the ciphertext policy to realize partial policy hiding. Later work (e.g. [17]–[19]) also realized the hiding of access control policy, solving data confidentiality protection problems, and fine-grained access control in cloud storage. The previous schemes are all based on the selective model. In the selective model, the adversary needs to select the access control policy to be challenged before the system initializes the public parameters. Lewko *et al.* [22] first solved this problem. They proposed a fully secure ABE system using the dual system encryption methodology introduced by Waters [23] and techniques used by Lewko *et al.* [24]. However, these encryption schemes do not support keyword search and cannot search data ciphertext.

In order to effectively support one-to-many application scenarios in cloud storage, Zheng *et al.* [13] and Sun *et al.* [14] combine attribute-based encryption (ABE) technology with searchable encryption (SE) technology. The attribute-based encryption with keyword search scheme was proposed, which supports the fine-grained access control of ciphertext data and provides fast retrieval of ciphertext data, effectively improving the sharing efficiency of ciphertext data in the cloud. In an ABKS system [15], [35], the cloud server cannot get any information about the keyword(s) and the data.

Qiu *et al.* [20] and Wu *et al.* [21] presented an ABKS scheme with hidden access policy, and their schemes can resist KGA. Liu *et al.* [28] also proposed an HPCPABKS scheme supporting integrity verification and data deduplication. However, most of the schemes were only proven secure in the selective model. The selective security model assumes that the adversary selects the access structure to attack before initialization. Therefore, to resist KGA, these selective security schemes [20], [21], [28] need to prove that the ciphertext is indistinguishable from the adversary first, then prove that the keyword trapdoor is indistinguishable from the adversary. However, the full security model is more secure. It has been proved in [33] that the keyword ciphertext of the ABKS scheme transformed by the full security ABE is indistinguishable from the adversary. Moreover, the trapdoor indistinguishability can be proved along with the security proof, which gives a stronger sense of integrity to the security proof. To effectively manage encrypted cloud email scenarios and improve their security, we designed a full security HPCPABKS scheme based on composite order bilinear group. Compared with several encrypted email PEKS schemes [29]–[31] introduced in section I, the ABKS scheme supporting multi-keyword or other types of keywords is more difficult to construct. Therefore, our scheme now only supports ordinary single keyword search.

Organization: In Section III, we give all the preliminaries, including complexity assumptions, access structure. We formally define our encrypted cloud email searching and filtering scheme and its security, and present the concrete scheme in Section IV. In Section V, we give the security proof of our scheme and comparison. In Section VI, we state our conclusion.

III. PRELIMINARIES

A. COMPOSITE ORDER BILINEAR GROUPS

Boneh *et al.* first introduced Composite order bilinear in [25]. Our scheme is constructed on the groups of order N, where N is the product of three distinct primes. Let δ be a group generator, an algorithm which takes a security parameter 1^λ as input and outputs a tuple (p, q, r, G, G_T, e). δ outputs (p, q, r, G, G_T, e) where p, q, r are distinct primes, G and G_T are cyclic groups of order $N = pqr$. $e : G \times G \to G_T$ is a map such that:

1) Bilinear. $\forall g, h \in G, \forall a, b \in Z_N, e(g^a, h^b) = e(g, h)^{ab}$
2) Non-degenerate. $\exists g \in G, e(g, g)$ has order N in G_T,
3) The multiplication in G and G_T, as well as the operations of bilinear maps e, are computable in polynomial time with respect to λ.

G_p, G_q and G_r denote the subgroups of group G whose order is p, q and r respectively. According to the orthogonality of subgroups, we can see that if $\forall h_p \in G_p$ and $\forall h_q \in G_q$ then $e(h_p, h_q) = 1$.

B. ASSUMPTION

Assumption 1 (Subgroup Decision Problem for 3 Primes [26]): δ is a generator of group G. We define the following distribution:

$$\begin{aligned}(p, \mathrm{q}, \mathrm{r}, G, G_T, e) &\leftarrow \delta(1^\lambda)\\ N &= pqr, \quad g_p \leftarrow G_p, \quad g_r \leftarrow G_r,\\ \mathrm{D} &= (G, G_T, \mathrm{N}, e, g_p, g_r),\\ T_1 &\leftarrow G_{pq}, \quad T_2 \leftarrow G_p\end{aligned}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 1 to be:

$$\mathrm{Adv}^1_{\mathcal{A}} |\Pr[\mathcal{A}(\mathrm{D}, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 1: We say δ satisfies Assumption 1 if for any polynomial time algorithm $\mathcal{A}$, $\mathrm{Adv}^1_{\mathcal{A}}$ is negligible.

Assumption 2: Given a group generator δ as above. We define the following distribution:

$$\begin{aligned}(p, \mathrm{q}, \mathrm{r}, G, G_T, e) &\leftarrow \delta(1^\lambda),\\ N &= pqr, g_p, \quad X_1 \leftarrow G_p, \quad X_2 \leftarrow G_q,\\ g_r &\leftarrow G_r,\\ \mathrm{D} &= (G, G_T, \mathrm{N}, e, g_p, X_1X_2, g_r)\\ T_1 &\leftarrow G_{pq}, \quad T_2 \leftarrow G_p\end{aligned}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 2 to be:

$$\mathrm{Adv}^2_{\mathcal{A}} |\Pr[\mathcal{A}(\mathrm{D}, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 2: We say δ satisfies Assumption 2 if for any polynomial time algorithm $\mathcal{A}$, $\mathrm{Adv}^2_{\mathcal{A}}$ is negligible.

Assumption 3: Given a group generator δ as above. We define the following distribution:

$$\begin{aligned}(p, \mathrm{q}, \mathrm{r}, G, G_T, e) &\leftarrow \delta(1^\lambda)\\ N &= pqr, \quad \alpha \in Z_N,\ g_p \leftarrow G_p,\ g_q,\\ Q, Q_1, Q_2 &\leftarrow G_q\\ g_r, R_0, R_1, R &\leftarrow G_r\\ \mathrm{D} &= (G, G_T, N, e, g_p R_0, g_P^\alpha R_1,\\ &\quad g_P Q_1, g_P^{1/\alpha} Q_2, g_q, g_r)\\ T_1 &= g_p^\alpha QR, \quad T_2 \leftarrow G_T\end{aligned}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 3 to be:

$$\mathrm{Adv}^3_{\mathcal{A}} | \Pr[\mathcal{A}(\mathrm{D}, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 3: We say δ satisfies Assumption 3 if for any polynomial time algorithm $\mathcal{A}$, $\mathrm{Adv}^3_{\mathcal{A}}$ is negligible.

C. ACCESS STRUCTURE

In the ABE system, access structure determines the expression ability of access control policy. There are mainly the following access structures: AND-gates, tree-based access structure, threshold structure, and linear secret sharing structure. In our scheme, we adopt AND-gates on multi-valued attributes [17], [20], [21] as our access structure. That is, AND-gate is used to connect different attributes, and OR-gate is used to connect different values of the same attribute. It does not support NOT-gate. For example:

Department: R & D AND Position: Programmer

AND (Seniority: Junior OR Seniority: Senior)

We denote an attribute list of one user as $L = \{L_1, L_2, \ldots, L_n\}$. We denote an access structure as $P = \{P_1, P_2, \ldots, P_n\}$. The attributes satisfy the access control structure, if and only if $L_i \in P_i$, $i \in [1, \ldots, n]$. This access structure is inspired by inner product encryption and can hide the access policy.

D. CPABE

A ciphertext-policy attribute-based encryption system consists of the following four algorithms: Setup, KeyGen, Encrypt, and Decrypt.

(1) *Setup* $\left(1^\lambda\right)$. This algorithm takes as input a security parameter λ and description of attributes. It generates a public key PK and a master secret key MSK.

(2) *KeyGen* (MSK, S). This algorithm takes the master secret key MSK, the public key PK, and an attribute list S as input. It generates a secret key SK associated with S.

(3) *Encrypt*(P, M, PK). This algorithm takes an access structure P, a message M, and the public key PK as input. It generates a ciphertext CT. It should be noted that the access policy of hidden policy CPABE is hidden in the ciphertext CT.

(4) *Decrypt*(CT, SK). This algorithm takes a ciphertext CT and a secret key SK as input. It returns a message M. If the attribute list S satisfies the access structure P specified for CT, the user can decrypt the ciphertext.

IV. ENCRYPTED CLOUD EMAIL SEARCHING AND FILTERING SCHEME

This section describes our encrypted cloud email searching and filtering scheme model, formal and security definitions, and the scheme's detailed process.

A. SCHEME MODEL

We illustrate the based architecture of our scheme in Figure 1. Our scheme involves five entities: Trusted authority, Sender, Email filtering server, Recipient, Cloud server. Each entity is introduced as follows:

1. Trusted authority: The Authorization center generates a public key and master key for the scheme and generates a private attribute key for the recipient filter server and all the email users (including sender and recipient). The user's private key is related to his attributes.

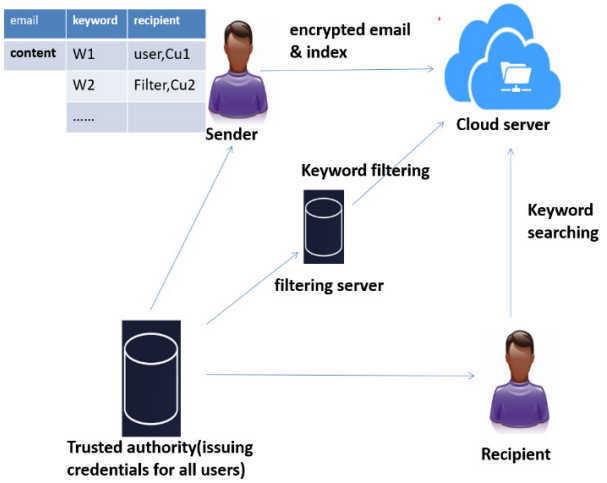


FIGURE 1. Basic architecture.

2. Sender: At first, the client of sender segments the contents of an email into keywords. The sender then creates an additional recipient list and adds all recipients and the corresponding filter server to this recipient list. The sender determines the access policy according to the attributes of the recipients and filter servers and uses the access policy to encrypt the keyword index. The email content is still encrypted with the original system's encryption method. Finally, the sender sends the ciphertext of the index and email to the cloud server.

3. Email filtering server: The email filter server maintains a blacklist of keywords to be filtered. It generates keyword trapdoors according to the blacklist and sends them to the cloud server to run the filtering algorithm to filter the newly received emails. The filter here actually searches.

4. Recipient: If the recipient wants to search for an email with a keyword in the recently received email, the recipient will generate a keyword trapdoor and then send the keyword trapdoor to the cloud server for searching. If the email contains keywords to search, the cloud server will submit the email to the recipient.

5. Cloud server: The encrypted emails with encrypted keyword index received by the recipient are stored on the cloud server. When the cloud server receives a trapdoor from the email filtering server or the recipient, it will provide keyword search or filter services. If the keyword matches, the cloud server will do the corresponding processing.

Our scheme does not provide the decryption of encrypted email. The original email encryption system provides the decryption of email. In the whole scheme, participants do not have to decrypt the encrypted email and encrypted keyword index, and the user's secret key will not be disclosed. In the whole process, the cloud server could not get any email content and keyword content.

We assume that the cloud server is semi-trusted (i.e., honest-but-curious). It means that the cloud server will dutifully store and query data, but it will be curious about the keyword indexes and search credentials and try to guess some information.

B. SCHEME DESCRIPTION

The scheme description consists of the following five algorithms: *Setup*, *KeyGen*, *Encindex*, *Trapdoor*, *Verify (Search/filter)*. The details are as follows:

(1). $MSK, PK \leftarrow Setup\left(1^{\lambda}\right)$: The trusted authority runs the *Setup* algorithm to initialize the system. It takes as input a security parameter λ and generates a public key PK and a master secret key MSK.

(2). $sk \leftarrow KeyGen\,(S, MSK)$: The trusted authority runs the *KeyGen* algorithm to generate the email user's secret key. It takes as input the MSK and an email user's attribute list S. Then the *KeyGen* algorithm generates a secret key *sk* associated with S.

(3). $CT \leftarrow Encindex(P, kw, PK)$: The sender runs the *Encindex* algorithm to encrypt the keyword index. It takes as input the system public key PK, the keyword *kw*, and the access structure *P*. Then, the *Encindex* algorithm generates the keyword *kw* ciphertext index *CT*.

(4). $trap \leftarrow Trapdoor(sk, tw)$: The recipient or email filter server runs the *Trapdoor* algorithm to generate a keyword trapdoor for searching or filtering. It takes a secret key *sk* and a query keyword or a filter keyword *tw* as input. Then the *Trapdoor* algorithm generates a keyword trapdoor *trap*.

(5). $0\ or\ 1 \leftarrow Verify(CT, trap)$: The cloud server runs the *Verify(Search/filter)* algorithm to search or filter the keyword index of email. It takes the keyword index ciphertext *CT* and a search or filter trapdoor *trap* as input. Then the *Verify(search)* algorithm outputs 1 when the recipient's attributes satisfy the access structure, and the trapdoor keyword matches the ciphertext keyword, otherwise 0. Similar to *Verify(search)* algorithm, the *Verify(filter)* algorithm is mainly used to match the email content keywords with the blacklist set. If the matching is successful, it will return the email identification 1 and do further processing. If the matching is unsuccessful, it will return 0.

C. SECURITY DEFINITION

(1). The full security model for our encrypted cloud email searching and filtering scheme is described by a security game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$. The *Game I* is defined as follows:

Setup. The Challenger $\mathcal{B}$ calls the $Setup\left(1^{\lambda}\right)$ algorithm to generate the master secret key MSK and the public key PK. Then the adversary $\mathcal{A}$ gets the PK, and the Challenger $\mathcal{B}$ saves the MSK.

Phase 1. The adversary $\mathcal{A}$ can make q_1 times private key queries according to attributes set $S_1, S_2, \ldots, S_{q_1}$. The challenger $\mathcal{B}$ generates the private key $sk_i \leftarrow KeyGen\,(S_i, MSK)$ for $1 \leq i \leq q_1$ and returns it to $\mathcal{A}$. The adversary $\mathcal{A}$ can also make query about keyword trapdoor. Then the challenger $\mathcal{B}$ runs $KeyGen\,(S_i, MSK)$ to generate sk_i and runs $Trapdoor(sk_i, tw_i)$ for $1 \leq i \leq q_1$ to get the keyword tw_i trapdoor and sends the trapdoor to the adversary $\mathcal{A}$.

Challenge. The adversary $\mathcal{A}$ selects two keywords w_0, w_1 and two access structures P_0, P_1 to send to the challenger $\mathcal{B}$. The challenger $\mathcal{B}$ chooses a random bit $\beta \in \{0, 1\}$, then generates challenge ciphertext $CT^* \leftarrow Encindex(P_\beta, w_\beta, PK)$. Finally, it returns CT^* to the adversary $\mathcal{A}$.

Phase 2. The adversary $\mathcal{A}$ repeats the queries of Phase1.

Guess. The adversary $\mathcal{A}$ gives its guess $\beta' \in \{0, 1\}$ for β. If $\beta' = \beta$, we say that the adversary $\mathcal{A}$ wins the game.

Definition 4: An encryption cloud email searching and filtering scheme is fully secure if all polynomial time adversaries have at most a negligible advantage in this security game, where the advantage of the adversary $\mathcal{A}$ is

$$\mathrm{Adv}^{f}_{\mathcal{A}} |\Pr[\beta' = \beta] - \tfrac{1}{2}|$$

(2). The game of keyword trapdoor indistinguishability is defined as follows:

Setup. Same as in *Game I*.

Phase 1. Same as in *Game I*.

Challenge. The adversary $\mathcal{A}$ selects two keywords w_0, w_1 and a challenge access structures P to send to challenger $\mathcal{B}$. The challenger $\mathcal{B}$ chooses a random bit $\beta \in \{0, 1\}$, then computes $trap \leftarrow Trapdoor(sk, w_\beta)$. Finally, it returns *trap* to the adversary $\mathcal{A}$.

Phase 2. The adversary $\mathcal{A}$ repeats the queries of Phase1.

Guess. The adversary $\mathcal{A}$ gives its guess $\beta' \in \{0, 1\}$ for β. If $\beta' = \beta$, we say that the adversary $\mathcal{A}$ wins the game.

Definition 5: An encryption cloud email searching and filtering scheme satisfies keyword trapdoor indistinguishability if all polynomial time adversaries have at most a negligible advantage in this security game, where the advantage of the adversary $\mathcal{A}$ is

$$\mathrm{Adv}^{t}_{\mathcal{A}} |\Pr[\beta' = \beta] - \tfrac{1}{2}|$$

D. THE CONSTRUCTION OF OUR SCHEME

Suppose the email user has n attributes, each of which has m possible values, and each attribute value corresponds to a different attribute item. Subgroup G_p is used for standard

encryption and verification operations; G_r is used for the randomization of parameters to make the scheme achieve full security; G_q is only used for a semi-functional space, not for the operation of the actual scheme.

(1). $MSK, PK \leftarrow Setup\left(1^{\lambda}\right)$: This algorithm first runs the group generator algorithm $\delta\left(1^{\lambda}\right)$ to get $(p, q, r, G, G_T, \mathrm{e})$ with $G = G_p \times G_q \times G_r$, where G and G_T are cyclic groups of order $N = pqr$. $H : \{0,1\}^* \rightarrow \mathbb{Z}_N$ is a hash function. It selects g_p be a generator of G_p and g_r be a generator of G_r. Next, it randomly selects $a_{i,j} \in \mathbb{Z}_N$ and $R_{i,j} \in G_r (1 \leq i \leq n, 1 \leq j \leq m)$. It computes $A_{i,j} = g_p^{a_{i,j}} R_{i,j}$. It also selects $\omega \in \mathbb{Z}_N$, $R_0 \in G_r$, and calculates $A_0 = g_p R_0$, $Y = e(g_p, g_p)^{\omega}$. Finally, it sets the public key PK $Pk = \{\mathrm{A}_0, \{\mathrm{A}_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq m}, g_r, Y\}$ and sets the master secret key MSK

$$MSK = \{g_p, \{a_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq m}, \omega\}.$$

(2). $sk \leftarrow KeyGen\,(L, MSK)$: Firstly, the trusted authority center randomly selects $x_u \in \mathbb{Z}_N$ for every new email user joining the system. Then it sets $X = e(g_p, g_p)^{\omega x_u} = Y^{x_u}$ and publishes X as a public parameter of the email user (including email filter server). The trusted authority runs this algorithm to generate the email user's secret key. It takes as input the $MSK = \{g_p, \{a_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq m}, \omega\}$ and an email user's attribute list $L = (v_{1,\mathrm{j}_1}, \ldots v_{i,\mathrm{j}_i}, \ldots, v_{n,\mathrm{j}_n}), \mathrm{j}_i \in \{1, \ldots, m\}$. This algorithm randomly selects $t_i \in \mathbb{Z}_N$ for $i = 1$ to n, and sets $t = \sum_{i=1}^{n} t_i$. Then it sets $D_0 = g_p^{\omega - t}$ and sets $D_i = g_p^{\frac{t_i}{a_{i,j_i}}}, 1 \leq i \leq n$. Finally, this algorithm generates the secret key $Sk = (x_u, D_0, \{D_i\}_{1 \leq i \leq \mathrm{n}})$

(3). $CT \leftarrow Encindex(P, kw, PK)$: The sender creates an additional recipient list and adds the filter server as the recipient. It calculates $C_U = X^{-s}$ for the recipient, and then a tuple $(recipientorfilter\ server, C_U)$ is generated for each recipient in this list. These users' attributes in this recipient list are used to generate the access control policy P.

The sender runs this algorithm to encrypt the email keyword index with the access structure P. Let $P = (w_1, w_2, \ldots, w_n), w_i \subseteq v_i$. This algorithm randomly selects $s \in \mathbb{Z}_N$ and $R'_0 \in G_r$. It also randomly selects $s_{i,j} \in \mathbb{Z}_N$ and $R'_{i,j} \in G_r, 1 \leq i \leq n, 1 \leq j \leq m$. It then sets $\tilde{C} = Y^s$ and also computes

$$C_{i,j} = \begin{cases} A_{i,j}^{s} \cdot R'_{i,j}, & if\ v_{i,j} \in w_i \\ A_{i,j}^{s_{i,j}} \cdot R'_{i,j}, & otherwise \end{cases} \quad 1 \leq i \leq n, \quad 1 \leq j \leq m$$

Here, $R'_{i,j}$ realizes the hidden policy by parameter randomization. In this way, the access policy is hidden in the ciphertext.

The content of the email is segmented by keyword kw. This algorithm computes $C_0 = A_0^{\frac{s}{H(kw)}} \cdot R'_0$. Lastly, it generates the ciphertext $CT = (\tilde{C}, C_0, \{C_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq m})$.

(4). $trap \leftarrow Trapdoor(sk, tw)$: The recipient or email filter server runs this algorithm to generate a trapdoor for a keyword tw. This algorithm first selects $d \in \mathbb{Z}_N$ and then computes $\tilde{T} = x_u + d$. Then it computes $T_0 = D_0^{H(\mathrm{tw})d}$ and $T_i = \{D_i\}_{1 \leq i \leq n}^{d}$. Finally, it generates the keyword trapdoor $Trap = (\tilde{T}, T_0, \{T_i\}_{1 \leq i \leq n})$.

(5). $0\ or 1 \leftarrow Verify(CT, trap)$: The cloud server checks whether the user is in the extra recipient list. If not, it declines the search request. Otherwise, the cloud server takes encrypted keyword index and keyword trapdoor as input and C_U from the additional recipient list to run the *Verify* algorithm. Then the cloud server can recover the intermediate value $E = \prod_{i=1}^{n} e(C_{i,j_i}, T_i)$ by combining C_{i,j_i} and T_i, and can search out the ciphertext matching keyword through this intermediate value. The cloud server calculates:

$$e(C_0, T_0) \cdot \prod_{i=1}^{n} e(C_{i,j_i}, T_i) = \tilde{C}^{\tilde{T}} \cdot C_U$$

It returns 1 if two sides are equal and return 0 if otherwise.

If the recipient or recipient filter server's attributes satisfy the access policy and the encryption keyword and the threshold keyword are equal, the left is equal to the right.

The correctness of the scheme is verified as follows. In the first step, if the user's attributes $L = (v_{1,\mathrm{j}_1}, \ldots v_{i,\mathrm{j}_i}, \ldots, v_{n,\mathrm{j}_n})$ satisfy the access control policy $P = (w_1, w_2, \ldots, w_n)$, according to the previous formula $C_{i,j} = A_{i,j}^{s} \cdot R'_{i,j} (if\ v_{i,j} \in w_i)$, then:

$$\begin{aligned} E &= \prod_{i=1}^{n} e(C_{i,j_i}, T_i) \\ &= \prod_{i=1}^{n} e(A_{i,j_i}^{s} \cdot R'_{i,j_i}, g_p^{\frac{dt_i}{a_{i,j_i}}}) \\ &= \prod_{i=1}^{n} e(g_p, g_p)^{t_i sd} \\ &= e(g_p, g_p)^{tsd} \end{aligned}$$

Second, if the search filtered keyword is the same as the keyword in the encrypted index $kw = tw$, the cloud server successfully passes the following validation and outputs 1.

$$\begin{aligned} e(C_0, T_0) \cdot E &= e(A_0^{\frac{s}{H(kw)}} \cdot R'_0, g_p^{H(\mathrm{tw})(\omega - t)d}) \cdot e(g_p, g_p)^{tsd} \\ &= e(g_p, g_p)^{(\omega - t)sd} \cdot e(g_p, g_p)^{tsd} \\ &= e(g_p, g_p)^{\omega sd} \\ \tilde{C}^{\tilde{T}} \cdot C_U &= Y^{s(x_u + d)} \cdot Y^{-sx_u} = Y^{sd} = e(g_p, g_p)^{\omega sd} \end{aligned}$$

V. SECURITY PROOF AND COMPARISON

A. SECURITY PROOF

We apply the dual system encryption to obtain full security. In our dual system, there is not only semi-functional ciphertext and semi-functional key but also another structure: semi-functional keyword trapdoor. The normal keyword trapdoor can verify normal ciphertexts or semi-functional ciphertexts. The semi-functional keyword trapdoor can verify normal ciphertext, but cannot verify semi-functional ciphertext. That is to say. A normal keyword trapdoor can only verify semi-functional ciphertext. Security proof of our scheme will also be demonstrated through a series of indistinguishable games. In the first game, all the ciphertexts and keyword trapdoors are normal. In the next game, all the keyword trapdoors are

TABLE 1. Game process.

GAME	Challenge ciphertext	Key and Trapdoor of query				
		1	.....	k	.....	q
Real	Normal					
0	Semi-Functional	Normal				
1	Semi-Functional		Normal			
.....						
k	Semi-Functional				Normal	
.....						
q	Semi-Functional					
Final	Random message	Semi-Functional				

normal, but the ciphertexts are semi-functional. We suppose that an adversary makes q keyword trapdoor queries. Then in $Game_k$, the first k keyword trapdoors are semi-functional, and the rest are normal. In $Game_q$, all the keyword trapdoors and ciphertexts returned to the adversary are semi-functional, so all the keyword trapdoors cannot verify the challenge ciphertext.

Firstly, we define three semi-functional structures:

Semi-functional Ciphertext We let g_q denote a generator of the subgroup G_q. A semi-functional ciphertext is generated as follows:

Firstly, run the *Encindex* algorithm to generate a normal ciphertext $CT = (\tilde{C}', C_0', \{C_{i,j}'\}_{1\le i\le n, 1\le j\le m})$. Then randomly select $x_0, x_{i,j} \in \mathbb{Z}_N, 1 \le i \le n, 1 \le j \le m$ and generate the semi-functional ciphertext:

$$CT = (\tilde{C}', C_0 = C_0' \cdot g_q^{x_0}, \{C_{i,j} = C_{i,j}' \cdot g_q^{x_{i,j}}\}_{1\le i\le n, 1\le j\le m})$$

Semi-functional Key A semi-functional key is generated as follows:

Firstly, run the *KeyGen* algorithm to generate a normal key $Sk = (x_u', D_0', \{D_i'\}_{1\le i\le n})$. Then randomly select $y_0, y_i \in \mathbb{Z}_N, 1 \le i \le n$ and generate the semi-functional key: $Sk = (x_u', D_0 = D_0' \cdot g_q^{y_0}, \{D_i = D_i' \cdot g_q^{y_i}\}_{1\le i\le n})$

Semi-functional keyword trapdoor. A semi-functional keyword trapdoor is generated as follows:

Firstly, run the *Trapdoor* algorithm to generate a normal keyword trapdoor $Trap = (\tilde{T}', T_0', \{T_i'\}_{1\le i\le n})$. Then randomly select $z_0, z_i \in \mathbb{Z}_N, 1 \le i \le n$ and generate semi-functional keyword trapdoor: $Trap = (\tilde{T}', T_0' \cdot g_q^{z_0}, \{T_i' \cdot g_q^{z_i}\}_{1\le i\le n})$

To prove our scheme's security, we define a series of simulation games between the adversary and the challenger. Then we use three lemmas to prove that the games described below are indistinguishable from each other. As mentioned earlier, if both the challenge ciphertext and keyword trapdoors are semi-functional, then the validation effect will be invalid. We suppose the adversary to make q private key and keyword trapdoor requests in the game and define the following games (As shown in Table 1):

$Game_{real}$: It is the actual security game. All the keyword trapdoors, keys, and ciphertext are normal.

$Game_0$: All the keyword trapdoors and keys are normal, but the ciphertext is semi-functional.

$Game_k$: The challenge ciphertext is semi-functional. The first $k(1 \le k \le q)$ keyword trapdoors and keys are semi-functional, and the rest are normal. In $Game_q$, all the keyword trapdoors, keys, and ciphertext are semi-functional.

$Game_{final}$: All the keyword trapdoors and keys are semi-functional. The ciphertext is semi-functional encryption on a random keyword.

Lemma 1: Suppose there exists an algorithm $\mathcal{A}$ which has advantage $\mathrm{Adv}_{\mathcal{A}}^{Game_{real}} - \mathrm{Adv}_{\mathcal{A}}^{Game_0} = \varepsilon$ *to distinguish $Game_{real}$ and $Game_0$, then we can build an algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 1.*

Proof: Algorithm $\mathcal{B}$ is given g_p, g_r, T. $\mathcal{B}$ will distinguish $T \in G_p$ or $T \in G_{pq}$. Firstly, $\mathcal{B}$ randomly selects $a_{i,j} \in \mathbb{Z}_N$ and $R_{i,j} \in G_r (1 \le i \le n, 1 \le j \le m)$. Then it computes $A_{i,j} = g_p^{a_{i,j}} R_{i,j}$. $\mathcal{B}$ also randomly selects $\omega \in \mathbb{Z}_N$ and $R_0 \in G_r$. $\mathcal{B}$ gives algorithm $\mathcal{A}$ the public parameters $\{g_p R_0, \{A_{i,j}\}_{1\le i\le n, 1\le j\le m}, g_r, e(g_p, g_p)^{\omega}\}$ and keeps the master secret key MSK $\{g_p, \{a_{i,j}\}_{1\le i\le n, 1\le j\le m}, \omega\}$.

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two keywords w_0, w_1 and two access structures P_0, P_1. To generate the challenge ciphertext, $\mathcal{B}$ randomly selects $\beta \in [0, 1]$ and encrypts w_β with access structure P_β. $\mathcal{B}$ randomly selects $s \in \mathbb{Z}_N$ and $R_0' \in G_r$. Then $\mathcal{B}$ sets $\tilde{C}^* = e(g_p, T)^{\omega s}$, $C_0^* = T^{\frac{s}{H(w_\beta)}} R_0'$. It also randomly selects $s_{i,j} \in \mathbb{Z}_N$ and $R_{i,j}' \in G_r, 1 \le i \le n, 1 \le j \le m$. If $v_{i,j} \in w_i$, Let $C_{i,j}^* = T^{a_{i,j} \cdot s} R_{i,j}'$, otherwise let $C_{i,j}^* = T^{a_{i,j} \cdot s_{i,j}} R_{i,j}'$. Finally, algorithm $\mathcal{B}$ sends the challenge ciphertext $\{\tilde{C}^*, C_0^*, \{C_{i,j}^*\}_{1\le i\le n, 1\le j\le m}\}$ to $\mathcal{A}$.

If $T \in G_p$, the challenge ciphertext is normal. If $T \in G_{pq}$, the challenge ciphertext is semi-functional. Algorithm $\mathcal{A}$ can distinguish $Game_{real}$ and $Game_0$ with advantage ε. If $\mathcal{A}$ considers $\mathcal{B}$ as simulating $Game_0$, the challenge ciphertext will be semi-functional, and $\mathcal{B}$ will judge $T \in G_{pq}$. If $\mathcal{A}$ considers $\mathcal{B}$ as simulating $Game_{real}$, the challenge ciphertext will be normal, and $\mathcal{B}$ will judge $T \in G_p$. Hence, algorithm $\mathcal{B}$ can break Assumption 1 using the output of $\mathcal{A}$.

Lemma 2: Suppose there exists an algorithm $\mathcal{A}$ such that $\mathrm{Adv}_{\mathcal{A}}^{Game_{k-1}} - \mathrm{Adv}_{\mathcal{A}}^{Game_k} = \varepsilon$. *Then we can build an algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 2.*

($1 \le k \le q$, when $k = 1$, the $Game_{k-1}$ is $Game_0$)

Proof: Algorithm $\mathcal{B}$ is given g_p, X_1X_2, g_r, T. $\mathcal{B}$ will distinguish $T \in G_p$ or $T \in G_{pq}$. Firstly, $\mathcal{B}$ randomly selects $a_{i,j} \in \mathbb{Z}_N$ and $R_{i,j} \in G_r (1 \le i \le n, 1 \le j \le m)$. Then it computes $A_{i,j} = g_p^{a_{i,j}} R_{i,j}$. $\mathcal{B}$ also selects

TABLE 2. Comparison with other schemes.

scheme	Encrypt	Trapdoor	Search	Filter function
[29]	$2E_1 + E_2$	$2E_1 + E$	$2P$	No
[30]	$2E_1 + E_2 + 2P$	$E_1 + E_2 + P$	$2E_1 + 2P$	No
[31]	$2E + P$	$3E$	$4P$	No
ours	$(2nm+1)E+E_T$	$(n+1)E$	$(n+2)E_T+(n+1)P$	YES

$\omega \in \mathbb{Z}_N$ and $R_0 \in G_r$. $\mathcal{B}$ gives $\mathcal{A}$ the public parameters $\{g_pR_0, \{A_{i,j}\}_{1\le i\le n, 1\le j\le m}, g_r, e(g_p, g_p)^{\omega}\}$ and keeps the master secret key MSK $\{g_p, \{a_{i,j}\}_{1\le i\le n, 1\le j\le m}, \omega\}$. When $\mathcal{A}$ requests the *ith* keyword trapdoor or key, $\mathcal{B}$ answers as follows:

If $i > k$, $\mathcal{B}$ will create the normal keyword trapdoor and key for the queried by $\mathcal{A}$.

If $i < k$, $\mathcal{B}$ will create the semi-functional keyword trapdoor and key for the queried by $\mathcal{A}$ using the method in the **semi-functional** structure described earlier.

To answer the *kth* requested. $\mathcal{B}$ sets the semi-functional key as $\{x'_u, D'_0 = T^{\omega-t}, D'_i = T^{\frac{t_i}{a_{i,j}}}\}$ and sets the semi-functional keyword trapdoor as $\{\tilde{T}', T'_0 = T^{(\omega-t)H(tw_i)d}, T'_i = T^{\frac{t_i d}{a_{i,j}}}\}$ according to the master key MSK. Then it sends the semi-functional key and semi-functional trapdoor to $\mathcal{A}$.

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two keywords w_0, w_1 and two access structures P_0, P_1. To generate the challenge ciphertext, $\mathcal{B}$ randomly selects $\beta \in [0, 1]$ and encrypts w_β with access structure P_β. $\mathcal{B}$ randomly selects $s \in \mathbb{Z}_N$ and $R'_0 \in G_r$. $\mathcal{B}$ then sets $\tilde{C}^* = e(g_p, X_1X_2)^{\omega s}, C_0^* = (X_1X_2)^{\frac{s}{H(w_\beta)}}R'_0$. It also randomly selects $s_{i,j} \in \mathbb{Z}_N$ and $R'_{i,j} \in G_r, 1 \le i \le n, 1 \le j \le m$. If $v_{i,j} \in w_i$, sets $C_{i,j}^* = (X_1X_2)^{a_{i,j}\cdot s}R'_{i,j}$, otherwise sets $C_{i,j}^* = (X_1X_2)^{a_{i,j}\cdot s_{i,j}}R'_{i,j}$. Finally, $\mathcal{B}$ sends the semi-functional challenge ciphertext $\{\tilde{C}^*, C_0^*, \{C_{i,j}^*\}_{1\le i\le n, 1\le j\le m}\}$ to $\mathcal{A}$.

If $T \in G_p$, the keyword trapdoor and key are normal. If $T \in G_{pq}$, the keyword trapdoor and key are semi-functional. Algorithm $\mathcal{A}$ can distinguish $Game_k$ and $Game_{k-1}$ with advantage ε. If $\mathcal{A}$ considers $\mathcal{B}$ as simulating $Game_{k-1}$, the keyword trapdoor and key will be semi-functional, and $\mathcal{B}$ will judge $T \in G_{pq}$. If $\mathcal{A}$ considers $\mathcal{B}$ as simulating $Game_k$, the keyword trapdoor and key will be normal, and $\mathcal{B}$ will judge $T \in G_p$. Hence, algorithm $\mathcal{B}$ can break Assumption 2 using the output of $\mathcal{A}$.

Lemma 3: Suppose there exists an algorithm $\mathcal{A}$ such that $\mathrm{Adv}_{\mathcal{A}}^{Game_q} - \mathrm{Adv}_{\mathcal{A}}^{Game_{final}} = \varepsilon$. *Then we can build an algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 3.*

Proof: $\mathcal{B}$ is given $g_pR_0, g_P^{\alpha}R_1, g_PQ_1, g_P^{1/\alpha}Q_2, g_q, g_r, T$. Algorithm $\mathcal{B}$ will distinguish $T = g_P^{\alpha}QR$ or $T \in G_T$. Firstly, $\mathcal{B}$ randomly selects $a_{i,j} \in \mathbb{Z}_N$ and $R_{i,j} \in G_r (1 \le i \le n, 1 \le j \le m)$. Then it computes $A_{i,j} = g_p^{a_{i,j}}R_{i,j}$. $\mathcal{B}$ also selects $\omega \in \mathbb{Z}_N$ and $R_0 \in G_r$. $\mathcal{B}$ gives $\mathcal{A}$ the public parameters $\{g_pR_0, \{A_{i,j}\}_{1\le i\le n, 1\le j\le m}, g_r, e(g_p, g_p)^{\omega}\}$.

Algorithm $\mathcal{B}$ returns the semi-functional keyword trapdoor and key requested by $\mathcal{A}$. for $\forall i \in \mathbb{Z}_k$, $\mathcal{B}$ chooses $t_i \in \mathbb{Z}_N$ at random, sets $t = \sum_{i=1}^{n} t_i$, computes the semi-functional key:

$$x'_u, \quad D'_0 = (g_p^{1/\alpha}Q_2)^{\omega-t}, \quad D'_i = (g_p^{1/\alpha}Q_2)^{\frac{t_i}{a_{i,j}}}.$$

It computes the semi-functional trapdoor:

$$\tilde{T}', \quad T'_0 = (g_p^{1/\alpha}Q_2)^{(\omega-t)H(tw_q)d}, \quad T'_i = (g_p^{1/\alpha}Q_2)^{\frac{t_i d}{a_{i,j}}}.$$

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two keywords w_0, w_1 and two access structures P_0, P_1. To generate the challenge ciphertext, $\mathcal{B}$ randomly selects $\beta \in [0, 1]$ and encrypts w_β with access structure P_β. $\mathcal{B}$ randomly selects $s \in \mathbb{Z}_N$, $R'_0 \in G_r$. Then $\mathcal{B}$ sets $\tilde{C}^* = e(g_p, g_p)^{s\omega}$, $C_0^* = T^{\frac{s}{H(w_\beta)}}R'_0$. It also randomly selects $s_{i,j} \in \mathbb{Z}_N$ and $R'_{i,j} \in G_r, 1 \le i \le n, 1 \le j \le m$. If $v_{i,j} \in w_i$, sets $C_{i,j}^* = (g_pQ_1)^{a_{i,j}\cdot s}R'_{i,j}$, otherwise sets $C_{i,j}^* = (g_pQ_1)^{a_{i,j}\cdot s_{i,j}}R'_{i,j}$. Finally, algorithm $\mathcal{B}$ sends the challenge ciphertext $\{\tilde{C}^*, C_0^*, \{C_{i,j}^*\}_{1\le i\le n, 1\le j\le m}\}$ to $\mathcal{A}$.

If $T = g_P^{\alpha}QR$, C_0^* could be written $(g_P^{\alpha}QR)^{\frac{s}{H(w_\beta)}}R'_0$, then the challenge ciphertext is a semi-functional ciphertext with w_β, and the game is $Game_q$. If $T \in G_T$, then the challenge ciphertext is a semi-functional ciphertext with a random keyword, and the game is $Game_{final}$. Hence, if $\mathcal{A}$ can distinguish which game it plays, then $\mathcal{B}$ could use the output of $\mathcal{A}$ to break Assumption 3.

Theorem 1: If assumption 1, 2, and 3 hold, the proposed encryption cloud email searching and filtering scheme is secure.

Proof: If assumption 1, 2, and 3 hold, then we have shown by the previous lemmas that $Game_{real}$ is indistinguishable from $Game_{final}$. But in $Game_{final}$, the value of β is information theoretically hidden from the adversary $\mathcal{A}$. Therefore, we can conclude that the adversary $\mathcal{A}$ cannot attain a non-negligible advantage in breaking the encryption cloud email searching and filtering scheme.

Theorem 2: If assumption 2 hold, then the trapdoor of the proposed scheme is indistinguishable from adversary $\mathcal{A}$.

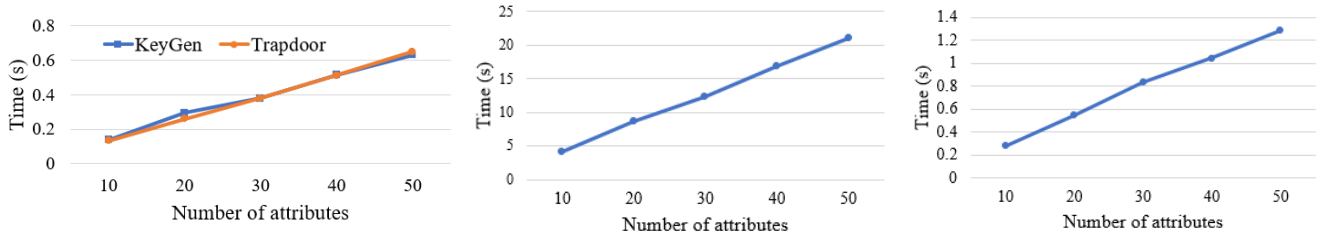
Proof: Assume the adversary $\mathcal{A}$ has an advantage ε to distinguish the keyword trapdoor.

According to TABLE 1 and Lemma 2, If $T \in G_p$, the keyword trapdoor is normal. If $T \in G_{pq}$, the keyword trapdoor is semi-functional. If adversary $\mathcal{A}$ has an advantage ε to distinguish the keyword trapdoor. The algorithm β can break assumption 2 with the same advantage.

If assumption 2 holds, adversary $\mathcal{A}$ can only distinguish the keyword trapdoor with negligible advantage. Therefore, the

TABLE 3. Real performance–average execution time (s) of each algorithm.

algorithm	the number of attributes				
	10	20	30	40	50
Setup	4.199	9.726	12.395	16.77	20.438
KeyGen	0.138	0.293	0.38	0.513	0.629
Encindex per keyword	4.163	8.707	12.315	16.846	21.037
Trapdoor	0.133	0.26	0.378	0.51	0.647
Verify/Search	0.279	0.543	0.837	1.044	1.287



(a)KeyGen and Trapdoor (b)Encindex per keyword (c)Verify/Search

FIGURE 2. Performance of our scheme.

keyword trapdoor of the proposed scheme is indistinguishable from adversary $\mathcal{A}$.

B. COMPARISON

We compare the performance of our scheme with related schemes [29]–[31]. The computational comparison is shown in Table 2. The P represents the pairing operation, E, E_1, E_2 and E_T represents the group exponentiation operations in G, G_1, G_2 and G_T. The user has n attributes, each of which has m possible values. In the algorithms, Encrypt represents encryption operation; Search represents verification operation on the server; Trapdoor represents generation trapdoor operation.

As shown in Table 2, traditional PEKS schemes [29]–[31] have less computation, but they don't provide email filtering. Their schemes need to generate an encryption index for each recipient when sending the mass mailing and encrypt multiple times when there are many recipients. Correspondingly, our scheme only needs to encrypt the index once. Compared with other schemes, we only add a recipient list, so the scheme is feasible. In terms of security, our scheme is under the full security model, making it more secure. Therefore, the keyword trapdoor of the scheme is indistinguishable from the adversary, and the scheme can resist KGA. In comparison, our scheme achieves functional improvement and security improvement at the cost of increased computing cost.

C. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed scheme, we conduct the experiments using real-world email dataset. The dataset used for the experiments is BC3-Email Corpus [36], which contains 40 emails and averaging five keywords per email. The experiments are performed with Intel Core i7-7700k 4.2GHz CPU, 16GB RAM, and Windows 10 operating system. The algorithms are implemented using JAVA programming language and the Java Pairing Based Cryptography Library (JPBC) [37] with A1 type pairing.

In the experiments, the security parameter is set to 128 bits, and the number of attributes increased from 10 to 50. To simplify the experiments, we set the number of possible values for each attribute to 20 and set all the keywords are encrypted under the same policy. Each experiment has been run ten times, and we take the average time as the final result. The average execution time of each algorithm in our scheme is shown in table 3.

As shown in Figure 2, the computation cost of each algorithm in our scheme is positively related to the number of attributes. We can also find that the cost of Encindex per keyword is far more than that of the algorithm Verify/Search. The main reason is that it costs much computation to implement policy hiding with composite order bilinear groups and create additional recipient list. Consistent with the previous theoretical analysis, our scheme improves the security and function at the cost of increasing the computation cost.

VI. CONCLUSION

We introduce a new solution for searching and filtering encrypted cloud email based on HPCPABKS. Our scheme enables the recipients to search for keywords and the recipient filtering servers to filter keywords by adding the additional list of recipients. The scheme achieves full security proved by using dual system encryption methodology and can resist offline KGA. It can be as convenient for users to search and filter as the traditional email system. More extensions can be made to the scheme to realize the function of virus email protection in the future. It can also be easily extended to other application scenarios, such as the searching and filtering encrypted file systems.

Because of the use of composite order bilinear groups, the performance of our scheme is limited. In the future, we need to improve the scheme further to make it more rapid and straightforward without reducing the security. In addition, our next work will also focus on multi keyword search and other query expression capabilities.

REFERENCES

[1] *Email Statistics Report, 2021-2025 Executive Summary*. Accessed: Mar. 3, 2021. [Online]. Available: https://www.radicati.com/wp/wpcontent/uploads/2020/12/

[2] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. CRYPTO*, vol. 2139, 2001, pp. 213–229.

[3] D. Boneh, G. Di Crescenzo, and R. Ostrovsky, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, Interlaken, Switzerland, 2004, pp. 506–522.

[4] Q. Tang and L. Chen, "Public-key encryption with registered keyword search," in *Proc. Eur. Public Infrastruct. Workshop*. Berlin, Germany: Springer, 2009, pp. 163–178.

[5] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: A new vision for public-key cryptography," *Commun. ACM*, vol. 55, no. 11, pp. 58–64, 2012.

[6] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. 24th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2005, pp. 457–473.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.

[8] N. Attrapadung and B. Libert, "Expressive key-policy attribute-based encryption with constant-size ciphertexts," in *Proc. 14th Int. Conf. Pract. Theory Public Cryptogr.* Berlin, Germany: Springer, 2011, pp. 90–108.

[9] J. Li, Q. Yu, and Y. Zhang, "Key-policy attribute-based encryption against continual auxiliary input leakage," *Inf. Sci.*, vol. 470, pp. 175–188, Jan. 2019.

[10] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, "Flexible and fine-grained attribute-based data storage in cloud computing," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 785–796, Jan. 2016.

[11] J. Li, W. Yao, J. Han, Y. Zhang, and J. Shen, "User collusion avoidance CP-ABE with efficient attribute revocation for cloud storage," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1767–1777, Jun. 2018.

[12] J. Li, N. Chen, and Y. Zhang, "Extended file hierarchy access control scheme with attribute based encryption in cloud computing," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 2, pp. 983–993, Apr./Jun. 2021.

[13] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proc. IEEE Conf. Comput. Commun.*, Toronto, ON, Canada, Apr. 2014, pp. 522–530.

[14] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," in *Proc. IEEE Conf. Comput. Commun.*, Toronto, ON, Canada, Apr. 2014, pp. 226–234.

[15] S. Wang, D. Zhao, and Y. Zhang, "Searchable attribute-based encryption scheme with attribute revocation in cloud storage," *PLoS ONE*, vol. 12, no. 8, Aug. 2017, Art. no. e0183459.

[16] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Proc. 6th Int. Conf. Appl. Cryptogr. Netw. Secur.*, New York, NY, USA, 2008, pp. 111–129.

[17] J. Lai, R. H. Deng, and Y. Li, "Fully secure ciphertext-policy hiding CP-ABE," in *Proc. 7th Int. Conf. Inf. Secur. Pract. Exper.*, Guangzhou, China, 2011, pp. 24–39.

[18] X. Li, "Efficient ciphertext-policy attribute based encryption with hidden policy," in *Proc. 5th Int. Workshop Internet Distrib. Comput. Syst.*, Melbourne, VIC, Australia, 2012, pp. 146–159.

[19] J. Lai, R. H. Deng, and Y. Li, "Expressive CP-ABE with partially hidden access structures," in *Proc. 7th ACM Symp. Inf., Comput. Commun. Secur.*, Seoul, South Korea, 2012, pp. 18–19.

[20] S. Qiu, J. Liu, Y. Shi, and R. Zhang, "Hidden policy ciphertext-policy attribute-based encryption with keyword search against keyword guessing attack," *Sci. China Inf. Sci.*, vol. 60, no. 5, May 2017, Art. no. 052105.

[21] A. Wu, D. Zheng, Y. Zhang, and M. Yang, "Hidden policy attribute-based data sharing with direct revocation and keyword search in cloud computing," *Sensors*, vol. 18, no. 7, pp. 2–17, 2018.

[22] A. Lewko, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," *Eurocrypt*, vol. 6110, pp. 62–91, Dec. 2010.

[23] B. Waters, "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 5677, S. Halevi, Ed. Berlin, Germany: Springer, 2009, pp. 619–636.

[24] A. Lewko and B. Waters, "New techniques for dual system encryption and fully secure HIBE wITH SHORT CIPHERtexts," in *Theory of Cryptography* (Lecture Notes in Computer Science), vol. 5978, D. Micciancio, Ed. Berlin, Germany: Springer, 2010, pp. 455–479.

[25] D. Boneh, E. J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *Theory of Cryptography*, vol. 3378, J. Kilian, Ed. Berlin, Germany: Springer, 2005, pp. 325–341.

[26] V. Goyal, A. Jain, and O. Pandey, "Bounded ciphertext policy attribute based encryption," in *Proc. 35th Int. Colloq. Autom., Lang. Program.*, 2008, pp. 1–5.

[27] J. Byun, H. Rhee, and H. Park, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in *Proc. Secure Data Manage.*, 2006, pp. 75–83.

[28] X. Liu, T. Lu, X. He, X. Yang, and S. Niu, "Verifiable attribute-based keyword search over encrypted cloud data supporting data deduplication," *IEEE Access*, vol. 8, pp. 52062–52074, 2020.

[29] Y. Zhang, Y. Li, and Y. Wang, "Efficient conjunctive keywords search over encrypted E-Mail data in public key setting," *Appl. Sci.*, vol. 9, no. 18, p. 3655, Sep. 2019.

[30] H. Li, Q. Huang, J. Shen, G. Yang, and W. Susilo, "Designated-server identity-based authenticated encryption with keyword search for encrypted emails," *Inf. Sci.*, vol. 481, pp. 330–343, May 2019.

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

[32] J. Chen, "Cloud storage third-party data security scheme based on fully homomorphic encryption," in *Proc. Int. Conf. Netw. Inf. Syst. Comput. (ICNISC)*, Apr. 2016, pp. 155–159.

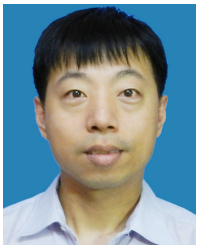
[33] F. Han, J. Qin, H. Zhao, and J. Hu, "A general transformation from KP-ABE to searchable encryption," *Future Gener. Comput. Syst.*, vol. 30, pp. 107–115, Jan. 2014.

[34] H. S. Rhee, W. Susilo, and H.-J. Kim, "Secure searchable public key encryption scheme against keyword guessing attacks," *IEICE Electron. Exp.*, vol. 6, no. 5, pp. 237–243, 2009.

[35] J. Li, X. Lin, Y. Zhang, and J. Han, "KSF-OABE: Outsourced attribute-based encryption with keyword search function for cloud storage," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 715–725, Sep./Oct. 2017.

[36] J. Ulrich, G. Murray, and G. Carenini, "A publicly available annotated corpus for supervised email summarization," in *Proc. AAAI Workshop*, 2008, pp. 77–82.

[37] *The Java Pairing Based Cryptography Library*. Accessed: Jun. 18, 2021. [Online]. Available: http://gas.dia.unisa.it/projects/jpbc/



JIAN GAO received the B.S. degree in computer science and technology from Shenyang Normal University, in 2000, and the M.S. degree in software engineering from Peking University, in 2007. He is currently pursuing the Ph.D. degree with the Software College, Northeastern University, China. His research interests include network security, information security risk assessment, and searchable encryption.

FUCAI ZHOU received the Ph.D. degree in computer software and theory from Northeastern University, China, in 2001. He is currently a Professor and a Ph.D. Supervisor with the Software College, Northeastern University. His research interests include cryptography, network security, trusted computing, secure cloud storage, software security, basic theory, and critical technology in electronic commerce.